This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problems Mailbox.

OPIC
OFFICE DE LA PROPRIÉTÉ INTELLECTUELLE DU CANADA

CIPO
CANADIAN INTELLECTUAL PROPERTY OFFICE

Ottawa Hull K1A 0C9

(21) (A1) 2,159,360
(86) 1994/06/29
(43) 1995/01/12

(51) Int.Cl.[6] H04L 9/30; E05B 49/00

(19) (CA) **APPLICATION FOR CANADIAN PATENT** (12)

(54) Method for Cryptographic Authentication of Transmitted Messages Using Pseudorandom Numbers

(72) Koopman, Philip J., Jr. - U.S.A. ;
Finn, Alan M. - U.S.A. ;

(71) United Technologies Automotive, Inc. - U.S.A. ;

(30) (US) 08/086,080 1993/06/30

(57) 46 Claims

Notice: This application is as filed and may therefore contain an incomplete specification.

Method For Cryptographic Authentication Of Transmitted Messages Using Pseudorandom Numbers

Technical Field

This invention relates to pseudo random numbers and cryptographically encoded transmissions, such as the type involved with an automobile key chain fob transmitter which opens the automobile door locks or trunk in response to transmissions from the fob.

Background Art

The art of encoding transmissions so that the transmissions may be authenticated at a receiving module must meet criteria for technical viability (security) as well as low cost and convenience. The cost and convenience criteria result in an inability to use any encoding with polynomials of excessive degree (such as binary numbers of hundreds of bits). Furthermore, cryptographic processing must require less than one second for acceptability by the user. Cost and weight constraints can limit the size and sophistication of a microprocessor or other signal processing equipment used in the system.

An example of such a system is disclosed in commonly owned U.S. Patent 5,191,610 to Hill and Finn. That system utilizes linear feedback shift register pseudo random number generation having the same seed number and the same, fixed feedback mask in the receiver as in the transmitter. The number of iterations of linear feedback shift register pseudo random number generation are counted in both the receiver and the transmitter, there being one additional iteration each time that a command is sent. Should the receiver not recognize one of the transmissions (because the transmitter was

inadvertently activated at a great distance from the receiver, or otherwise), the receiver is allowed a moderate number of catch-up iterations in which it attempts to match the received transmission. Should that fail, the transmitter tells the receiver how many iterations from the seed it should perform in order to recreate a new current pseudo random number in order to resynchronize the receiver to the transmitter pseudo random number.

The aforementioned system requires that a receiver and a transmitter be wired or loaded with a binary feedback mask at the factory and sold as a pair. It also precludes matching a replacement transmitter with an existing receiver without the involvement of dealership personnel, which could compromise security. The pseudo random number generators of the Hill and Finn patent use one iteration per encrypted message. This saves time but results in a certain level of correlation between successive samples, so that the samples are less random-like. In other pseudo random number applications, the speed advantage of the aforementioned system could be useful but for the inherent correlation.

Any such system, except one that uses a truly random number of infinite degree, can be compromised either by analysis of a succession of intercepted signals, or by a brute force, exhaustive numerical trial approach which simply tries every number possible as the authentication word (the code or key).

Coded keypads used for unlocking vehicles have inherent security features. The generation of the code word by pressing keys can be shielded from view, and is certainly not capable of being determined beyond a line of sight. Furthermore, there would be

great risk for an intruder entering every possible number into a keypad in an attempt to replicate the code (unless, of course, the automobile were parked in an unobservable area, such as a private or otherwise vacant garage). Thus, the keypad cannot be breached by analysis, and is not likely to be breached by numerical trial.

In contrast, lock systems which employ remote transmissions are enormously subject to security tampering because the surveillance of the transmissions may be carried out in another vehicle, without attracting any attention whatsoever. Therefore, it is possible to record many transmissions to a given vehicle, such as in a reserved workplace parking space (which commonly contains expensive cars), as well as providing an unobservable opportunity to attempt the breach of a security system (or even several systems at one time) by broadcasting huge volumes of random numbers, in parking lots where vehicles remain for long periods of time, such as at airports.

Whenever a transmitter is newly assigned to be used with an existing receiver, it is not sufficient to allow the new fob to identify itself and become authorized, without limiting that activity to a time when there is authorized access to the receiver through other than the transmitter itself (that is, within the vehicle itself). Thus, access to the vehicle by means of a traditional key or the like assures the safety of matching a newly assigned transmitter to an existing receiver. In the case of loss of synchronization between the transmitter and the receiver, simply allowing the receiver to synchronize to a particular pseudo random number provided thereto by the transmitter makes it too easy

for a surreptitious breach of security based on the analysis of a few transmissions, and synchronizing thereafter to one of the previous transmissions, utilizing numbers expected to be successful based upon analysis. Mere obfuscation of the resynchronizing code could be compromised by analysis of successful resynchronizations, and determination of the obfuscation function. The danger is not just that a single car might be broken into, but that a sophisticated capability might be developed and thereafter utilized extensively to breach the security of a large number of automobiles of a similar type.

Disclosure of Invention

Objects of the invention include provision of an improved remote operating system, the security of which is extremely difficult to breach by analysis, in which analysis of transmitted signals provides essentially no assistance in reducing the amount of numbers required for a numerical trial breach of security, and in which numerical trial breach of security requires, at a minimum, a prohibitively long time, rendering the vehicle essentially secure to brute force numerical trial attack, and which is useful only on a per vehicle basis. Other objects include rapid pseudo random number generation with minimal correlation.

This invention is predicated on our observation that introducing non-linearities into the Galois field operation of linear feedback shift register pseudo random numbers can render a code very difficult to breach by or with aid from numerical analysis. The invention is further predicated on the fact that time constraints on authentication can render the numerical trial approach essentially useless. The invention is

predicated in part on the reversibility characteristic of the well-known exclusive OR operation, and on the reversibility of encryption such as encryption involving linear feedback shift register operations.

According to the present invention, an encryption, such as a linear feedback shift register pseudo random number generation operation, is performed on a word comprising a pair of concatenated, independently generated numbers, which may themselves be encrypted (such as pseudo random numbers) and the result transmitted to a receiving module where a decryption, such as a reverse pseudo random number generation operation, recovers the concatenated numbers for cryptographic authentication. In accordance further with the invention, the encryption and decryption are performed with a secret mask essentially unique to the transmitter. According further to the invention, an encrypted number, such as a pseudo random number, used for cryptographic authentication contains command bits exclusive ORed into at least a portion thereof.

According to the invention, a number utilized in authentication of command transmissions is generated by an iterative encryption process, such as a linear feedback shift register pseudo random number generation operation, which has a variable number of iterations per authentication, the number varying in response to a pseudo random event. According further to the invention, a pair of pseudo numbers are iteratively encrypted, such as by linear feedback shift register pseudo random number generation operations, using a different number of iterations in each successive encryption, the sequence of the number of iterations of one of them being different from the sequence of the number of iterations of the other of

them, said number of iterations being based upon respectively different pseudo random events related to the respective words.

According to the invention, a plurality of transmitters are usable with a single receiver by means of secret numbers (such as initial values and feedback masks for feedback shift register encryption) essentially unique to each transmitter which are replicated in any receiver with which a transmitter is to be used, the receiver being capable of determining if each received transmission can be authenticated utilizing the initial values and masks of any of its assigned transmitters.

In accordance with the present invention, the initial secret values and feedback masks of transmitters to be associated with a given receiver are stored (such as in an erasable read-only memory) by downloading the secret numbers of a transmitter into a receiver, thereby permitting the addition of a transmitter to the family of transmitters to which a receiver can respond. Although this may be performed at a dealership, it does not require, nor need it permit, human knowledge of the precise numbers; therefore, the possibility of surreptitious access is nearly eliminated. In further accord with the invention, identification numbers are associated with the secret numbers of each transmitter, which pre-screen transmissions before authorization attempts, thereby reducing nuisance operations from similar, unauthorized transmitters, and reducing the time required for multiple authentication attempts.

According to the invention, the reception of each word causes the receiver of a remote operating system to become unresponsive to further transmissions for a period of about one-half second or more, thereby

significantly inhibiting the capability to breach the security thereof through exhaustive numerical trials.

According to the invention, cryptographic authentication of transmissions from a remote transmitter to a receiver module involves encryption, such as by linear feedback shift register pseudo random number generation operations, utilizing secret feedback masks which are essentially unique to each transmitter, and replicated only in a receiver which is to respond to the related transmitter.

Random iterations of the invention reduce correlation between successively generated pseudo random numbers. Cooperative use of all of the aforementioned features in a remote operating system renders the system extremely difficult to breach by analysis; and use of a half-second delay, for example, causes a 50% statistical probability of a 39 bit encrypted key word being breached by exhaustive numerical trials to require more than one month.

The invention may be used in remote systems other than automobile lock systems.

Other objects, features and advantages of the present invention will become more apparent in the light of the following detailed description of exemplary embodiments thereof, as illustrated in the accompanying drawing.

Brief Description of the Drawings

Fig. 1 is a stylized, simplified schematic block diagram of a transmitter according to the invention.

Fig. 2 is a stylized, simplified schematic block diagram of a receiver according to the invention.

Fig. 3 is a logic flow diagram of a transmitter encryption routine according to the invention.

Fig. 4 is a logic flow diagram of a switch interrupt subroutine which may be utilized in the present invention.

Fig. 5 is a logic flow diagram of a first portion of a receiver decryption routine according to the invention.

Fig. 6 is a logic flow diagram of a panic command or resynch command portion of a decryption routine according to the invention.

Fig. 7 is a logic flow diagram of a normal command authentication portion of a decryption routine according to the invention.

Figs. 8 and 9 are logic flow diagrams of alternative routines for pseudo random iterations.

Best Mode for Carrying Out the Invention

One example of the present invention is its use in a remote, encrypted automobile door and trunk locking and unlocking mechanism. The command to perform a certain task, such as a lock-related command (lock or unlock the doors, release the trunk), or to operate the lights and the horn or other alarm on the automobile in the event of a panic situation, or to cause cryptographic synchronization or resynchronization between the transmitter and the receiver module in the automobile, are under control of a plurality of buttons 12-15 disposed on a keychain fob or other hand-held transmitter unit shown in Fig. 1. The switches 12-15 may be tactile or touch-type and feed a microprocessor 17 which is associated with a PROM 19, a RAM 20 and a ROM 21. The PROM 19 is programmable only once, capable of having one set of output/input relationships burned therein, as is well known in the art. Typically, upon manufacture, each fob 16 will have its PROM burned in so as to establish

a 16 bit identification number, which is not protected as secret information, as is described more fully hereinafter; two secret initial (seed) values from which encrypted messages are originated; and three secret feedback masks defining suitable polynomials for feedback exclusive ORing in the encryption process, for the life of the fob, all as is described hereinafter. The RAM 20 is used as a scratch pad memory, in the usual way, and will contain changing values of the shift registers and commands, as described hereinafter. The program for the microprocessor 17 is contained in the ROM 21.

It is assumed that the microprocessor 17 is the type which has a stop mode in which the clock does not run, and the only function that the microprocessor can perform is to respond to an external interrupt, which in this case would be the closure of one of the switches 12-15. This keeps power consumption extremely low, and a suitable battery 24 could last about five years. The microprocessor 17 assembles a 64 bit command request word 25 which is applied serially over a line 26 to a suitable transmitter 27 (e.g., RF or infrared), which serially transmits the command request, as digital bits or otherwise, a suitable distance, such as not more than 10 meters. The fob 16 will, before beginning its useful life, be associated with a particular automobile along with up to three more fobs (in the example herein) so as to form a set of up to four fobs, any one of which can operate the locks or the panic alarm of a related receiver module 30 in an automobile or other secured enclosure. The receiver module 30 in the automobile includes a receiver 31 which receives the serial bits and applies them over a line 32 to a microprocessor 33, where the 64 bit word 25 is replicated in a 64 bit

word 38. The microprocessor 33 is powered from the automobile battery system 39. The microprocessor 33 has an electrically erasable PROM 40, a RAM 41 and a ROM 42 associated therewith. Each fob 16 is associated with a module 30 at a dealership, so that lost fobs may be replaced and matched to the module 30 anytime. A 64 bit word 25 (Fig. 1) is formulated with each fobs' ID, secret initial values and masks, and a download signal is provided, in some fashion, by factory personnel on a line 49. This may be achieved by a jumper, or in any other suitable way, since it does not pose a security threat unless the receiver 30 is tampered with simultaneously, which can be avoided as described below. The 64 bit word 25 sent to the receiver module 30 during a download includes one bit indicating the download operation. The presence of the download bit in the 64 bit word 38 (Fig. 2) can result in a download signal on a line 50 provided that the auto receiver 30 has been put into a download condition, such as by the installation of a download jumper 51 or other security measure. When download is suitably indicated, the fob ID and two initial values from the PROM 19 will be stored in the electrically erasable PROM 40. Then two secret feedback masks, of the same bit length as the initial values, will be sent with the ID in a similar fashion; and finally, a feedback mask which is as long as the concatenation of the two initial values is sent from the PROM 19 to the EE PROM 40 in the auto receiver 30. In a similar fashion, the initial values, feedback masks and ID's of three other fobs (in this example) will be loaded into the auto receiver 30 during valid download operations.

The word 25 appears in Fig. 1 to be within a special 64 bit register. However, the word 25

actually appears in various parts of the RAM 20, in addresses designated to be used for holding the parts of the outgoing, serially transmitted word. Similarly, all of the apparent hardware within the microprocessor 17 is merely illustrative of processes and relationships, which may indeed be performed by hardware which resembles that illustrated in Fig. 1, or may be implemented, as is preferred, by processing of bits utilizing the RAM 20 as a scratch pad memory, by means of software which is well within the skill of the art in the light of the teachings which follow hereinafter.

The microprocessor 33 has functions and processes illustrated therein which may either be hardware or software, as described with respect to Fig. 1 hereinbefore.

The narrative with respect to Fig. 1 is of a form describing hardware: software implementation of the invention is described with respect to Figs. 3-7, hereinafter.

The receiver module 30 is connected to the locks 43 of the automobile, as well as to the automobile horn and lights 44, or other suitable alarm arrangements on the automobile.

Before a fob can be utilized to operate the locks or alarms on an automobile, synchronization must occur. Herein, this is also referred to as resynchronization since it is utilized at different times during the life of the system, as described hereinafter. This resynchronization process is described hereinafter; suffice it at this point to say that the process will begin with the two secret initial (seed) values for the fob 16 located in a 20 bit linear feedback shift register (LFSR) 53, and a 19 bit LFSR 54, and suitable feedback masks for each of

the LFSRs 53, 54 available at the input of corresponding feedback exclusive ORs 55, 56. The initial synchronization (an initial resynchronization command) includes 20 iterations of the shift register 53 and at least 19 iterations of the shift register 54, so as to provide a complete bit-wise convolution. For ease in programming, both shift registers may be provided with 20 iterations during initialization (which is assumed herein). In each cycle, the high order bit is transferred by a line 61, 62 to the low order bit and is also exclusively ORed with those bits of the shift registers 53, 54 identified by bits in the feedback masks, to form the next higher order bits within the shift registers 53, 54. This is the very well known function of linear feedback shift registers, in the process of generating maximal length pseudo random numbers, as described in Numerical Recipes, Press, Flannery, Teukolsky, and Vetterling; Cambridge University Press, Cambridge, MA (1986). The feedback mask has to represent a suitable polynomial so as to provide a maximal length code, having degree N, which repeats only after 2^N-1 iterations. This is more fully set forth at pages 108-109 of Error Correcting Techniques for Digital Communication, Michelson and Levesque, John Wiley & Sons, New York, NY (1985), and in Appendix C of Error Correcting Codes, Pederson and Weldon, MIT Press, Cambridge, MA (1972).

One embodiment herein uses linear feedback shift registers, in some cases modified to be non-linear feedback shift register systems by shifting a pseudo random number of iterations as described hereinafter. However, other well-known pseudo random number generation techniques may be used such as linear congruential pseudo random number generators or non-

linear congruential pseudo random number generators as more fully set forth in Chapter 3 of The Art of Computer Programming, Volume 2/Seminumerical Algorithms, ed. 2, Knuth, Addison Wesley, Reading, MA (1981); or inverse congruential pseudo random number generators or generalized feedback shift register pseudo random number generators as more fully set forth in Chapters 7, 8 and 9 of Random Number Generation and Quasi-Monte Carlo Methods, Niederreiter, SIAM, Capitol City Press, Montpelier, VT (1992); or multiplicative, I/P, power, discrete exponential, kneading map, shift register, or cellular automatic pseudo random number generators as more fully set forth in "Pseudo random Number Generators in Cryptography and Number Theory", J. C. Lagarias, pages 115-143 of Cryptology and Computational Number Theory, Pomerance, ed., Volume 42, Proc. SIAM (1990).

In general the pseudo random number generator need not be reversible. A reversible generator is one where, given the current pseudo random number and complete knowledge of the generation process, the previous pseudo random number may be determined. For example, a linear feedback shift register pseudo random number generator is reversible.

The manner in which the system is originally synchronized and the registers are made ready to operate is described hereinafter, because the general operation should be first understood.

In the usual case, when everything has been established and the system is operating normally, assume that a lock, unlock or trunk release command has been provided by pressing one of the buttons 12-15. This will cause the microprocessor 17 to wake up and perform one cycle of operation. In the cycle of operation, the RAM 20 provides the values which were

previously left in the shift registers 53, 54 and the PROM 19 provides the masks for the shift registers 55, 56. Then, depending upon some pseudo-random event, such as the status of one or more bits of the shifts registers 53, 54, each of the shift registers will be provided one or two iterations or two or three iterations of linear feedback shifting due to the effect of clock gates 64 on the output of a clock 65. This is a first aspect of the present invention: instead of being shifted a number of times equal to the number of bits (which takes too many cycles to permit 256 attempts at decryption), the shift registers are only put through a few iterations after the initialization. Because this provides less scrambling of the feedback bits, the difficulty of mathematically ascertaining what the code might be is increased by causing the LFSRs to each undergo a different, variable number of iterations, in successive cycles, in a pseudo-random fashion. The pseudo random number provided by the iterations of the shift register 53 is supplied over a trunk of 20 lines 68 to a 39 bit shift register 69. The shift register 69 is associated with feedback 70 in the same fashion as the LFSRs 53, 54, with the exception that the shift register 69 is loaded with new numbers before each cycle of shifting feedback iterations. In this sense, then, the shift register 69 and feedback operate more as a cyclic redundancy code generator. The other input to the 39 bit shift register 69 is a trunk of 19 lines 73 from a gate 74 that causes the low order 5 bits of the 19 bit LFSR 54 on a trunk of lines 75 to be exclusive ORed with 5 bits on a trunk of 5 lines 76 from a command register 77. The command register 77 simply registers up to 32 commands encoded from the operation of any of the switches 12-15 (or fewer

commands if some bits are used in a discrete fashion). Thus, in each cycle, there is presented to the 39 bit shift register 69 the outputs of the LFSRs with a command exclusive ORed in the low order bits of one of them. Then, the shift register 69 undergoes 39 iterations of LFSR-type feedback through an exclusive OR process 70, which utilizes a secret feedback mask provided by the PROM 19. This provides a full bit-wise convolution of the two words from the shift registers 53, 54, which is a cryptographic necessity. Use of the shift register 69 may be employed in prior art systems, such as the two-generator embodiment of the Hill and Finn patent. When the 39 iterations are complete, the result is an encrypted, key word provided on a trunk of 39 lines 80 to the 64 bit word 25, along with 16 fob ID bits from the PROM 19, a download bit 49 if appropriate, and a command flag such as a panic/resynch bit provided from the command register 77 on a line 81, when appropriate. In the usual case of authentication, both of the download and panic/resynch bits will be 0's. Then, all of these bits are monitored on a trunk of 57 lines 82 by an error correcting code circuit 83 to create a 7 bit error correcting code component on a trunk of 7 lines 84 for the 64 bit word 25; typically, a single error correcting, multi-error detecting code (such as a Hamming code) will be used. The illustrated embodiment of the invention uses a linear feedback shift register as a cyclic redundancy code generator for encrypting the input into a key word. However, any of several well-known reversible encryption techniques may be used. For instance, the McEliece error correcting code encryption; the RSA cryptosystem; discrete exponentiation cryptosystem; linear or non-linear, full length or truncated

congruential cryptosystems; or the DES cryptosystem, as more fully set forth in Chapter 10 of Contemporary Cryptology: The Science of Information Integrity, Simmons, ed., IEEE Press New York, NY (1992).

When the 64 bit word is fully assembled, it is transmitted serially (bit-by-bit) or otherwise, by any well-known technique, through the transmitter 27 to the receiver 31 of the receiver module 30 to become the 64 bit word 38 therein. All of the bits of the word 38 are applied over trunks of 57 lines 90 and 7 lines 91 to an error correcting and detecting process 92. If a single bit error has occurred, a signal on a line 93 (as appropriate) will correct the bit that is in error. If a multiple bit error is detected, the process is totally void, and the receiver module 30 simply goes into a half second wait state, which simply slows down any attempts to crack the code which is being used, as is described more fully hereinafter. If a multiple bit error has occurred but is not detected, the cryptographic authentication process will almost certainly fail. On the other hand, if the error correcting code shows that the 64 bit word 38 has no errors, then a first OK signal is provided on a signal line 94.

When it is believed that there are no errors in the 64 bit word 38, it is proper to determine whether the 16 bit, non-secret identification word matches any of the fobs that have been loaded into the receiver module 30. The ID of the fob reduces the probability that a command from a wrong fob will be cryptographically acceptable; it also reduces the amount of time it takes to iterate the code words in the receiver module to reach authentication (a match). However, in this embodiment, there is no restriction on which fobs are assigned as a group to an

automobile, and it is assumed that there is approximately one chance in 11,000 that two fobs assigned to a particular automobile will have the same ID number. A feature of the invention is that if one fob with matching ID does not become authenticated, the receiver module 30 will see if there is another assigned fob with that same ID number, and if so, attempt authentication.

The 16 bit ID in the 64 bit word 38 is provided over a trunk of 16 lines 97 to a 16 bit compare circuit 98, the other inputs of which, on a trunk of 16 lines 99, are provided by the ID register 100, which really represents four different locations in the EE PROM 40, one for each associated fob. If, indeed, the message has come from one of the four associated fobs, a second OK signal appears on a line 101, and the identification number of the fob which has sent the message is provided on a trunk of 4 lines 102 to the PROM 40 and to the RAM 41 so as to utilize in the ensuing decryption process the secret mask for the selected fob and the two LFSR values which have previously been created for that fob. The previous LFSR values are utilized, rather than the initial secret values, because, according to the invention, the LFSR values are built upon, with only one, two or three iterations for each command received by the receiver module 30.

At this stage, the normal decryption process can begin. The 39 bit encrypted key word is provided over a trunk of 39 lines 107 to a 39 bit shift register 108 which can be identical in either structure or function to the 39 bit shift register 69 in the fob, except that it is iterated in a reversing process. The reversing process is easily understood, one bit at a time, by considering how the received 39 bits got to

be what they were. In the last iteration between the shift register 69 and the exclusive OR circuit 70 (Fig. 1) if the high order bit (leftmost bit in Fig. 1) was a 1, then exclusive ORing in accordance with the secret mask is provided against each bit of given order in the 39 bit shift register in order to determine what the next bit in order would be at the end of the iteration. That is to say, the ninth bit becomes the tenth bit (0 or 1 as the case may be) unless it is inverted by the exclusive OR. In order to be inverted by the exclusive OR, the ninth bit of the secret mask would have to be 1, and the most significant bit at the start of the iteration would also have to have been 1; and the most significant bit advances to the least significant stage, in a wrap-around. If either the most significant bit is a 0 or the corresponding bit in the secret mask is a 0, the ninth bit would simply advance into the tenth stage. Since what was the most significant bit becomes the least significant bit, inspection of the least significant bit determines whether or not exclusive ORing occurred. If the least significant bit in the shift register 108 is a 1, it is applied to exclusive OR the bits of each order with the same secret mask which was downloaded for this fob originally. For any bit (such as the ninth bit) for which there is a corresponding bit in the secret mask, whenever the lowest ordered bit at the start of the iteration is a 1, that bit will be inverted from 1 to 0 or from 0 to 1. But if there is no corresponding bit in the secret mask, then the bit in question is simply advanced to the next lower order stage (in the example here, bit 10 becomes bit 9) without being inverted. Or, if the least significant bit (the rightmost bit in Fig. 2) is a 0, then none of the bits are inverted as they are

advanced from one stage to the next lower stage in the shift register 108. By doing this the same number of times (39 iterations in the example herein), the original word in the 39 bit shift register 69 is reconstructed. The operation of the 39 bit shift register is very much like cyclic redundancy code (CRC) generators, used for error detection and correction. The process in the 39 bit shift registers herein is the same as in the LFSRs with the exception of the fact that the shift registers herein receive a whole new starting word before the iterations of each cycle. More on CRCs, Galois field arithmetic, and the generation and utilization of pseudo random binary numbers, may be found in Theory and Practice of Error Control Codes, Blahut, Addisson Wesley Pub. Company, Reading, MA (1984); An Introduction to Error-Correcting Codes, Shu Lin, Prentice Hall, Englewood Cliffs, NJ (1970); and Error-Control Techniques for Digital Communication, Michaelson and Levesque, John Wiley & Sons, New York, NY (1985).

In decryption, part of the process is reversed, and part of it is matched. Thus, the 39 bit encrypted code word is reversed by 39 reversing iterations, and the results thereof are compared to what should be identical results from the LFSRs.

Once a fob is identified in the 16 bit compare circuit 98, its two secret feedback masks are loaded (from RAM 41) for use in corresponding exclusive ORs 113, 114, and its previously achieved 20 bit LFSR value is loaded into a 20 bit LFSR 115, while its previously achieved 19 bit LFSR value is loaded into the 19 bit LFSR 116. Dependent upon a given bit of each of the LFSRs, the LFSR is shifted (with or without exclusive ORing as described hereinbefore) either once or twice, in the case of the LFSR 115 or

two or three times in the case of the LFSR 116 in dependence upon a pair of corresponding gates 117 which control the application of a clock 118 thereto, in the same fashion as described with respect to Fig. 1 hereinbefore. The 20 bits of the 20 bit LFSR 115 so generated are applied over a trunk of 20 lines 123 to a compare circuit 124, to be compared with 20 bits provided from the 39 bit shift register 108 over a trunk of 20 lines 125. Similarly, the high order 14 bits which are generated in the 19 bit LFSR 116 are provided by a trunk of 14 lines 127 to the compare circuit 124 for comparison with 14 bits of the 39 bit shift register 108 provided on a trunk of 14 lines 128. Assuming that both the 20 bit and 14 bit words compare properly, this signals a successful authentication on a line 129 and the receiver module 30 is allowed to receive and respond to the command made by the fob.

Recalling that the five bit command is exclusively ORed to the low order five bit positions provided from the 19 bit LFSR 54, the only way to recover those bits is to exclusive OR the low order 5 bit positions from the 19 bit LFSR 116 with the low order 5 bit positions of the reconstituted word in the 39 bit shift register 108. Therefore, the low order 5 bit positions produced by the 19 bit LFSR 116 are provided over a trunk of five lines 130 to a five bit exclusive OR circuit 131, the opposite inputs of which consist of the lowest order 5 bit positions from the 39 bit shift register 108 on a trunk of 5 lines 132. The result of the exclusive OR on a trunk of 5 lines 137 comprise the command which is stored in a command register 138. The typical commands provided on a trunk of lines 139 to the locks 43 comprise door unlock, door lock, and trunk release. Another command

indicated by a signal on a line 140 may comprise a panic command which will cause the horn and lights 44 (or other alarms) on the car to scare away a loiterer as the driver approaches the car with the fob (as described more fully hereinafter).

If the first attempt to match the outputs of the LFSRs 115, 116 with corresponding 34 bits of the 39 bit shift register 108 fails, then the LFSRs 115, 116 are cycled again. In each cycle, the LFSR 115 will be shifted once or twice depending upon the random bit utilized as a control over its gate 117, and the LFSR 116 will be shifted two or three times in dependence on the random bit utilized to control its clock gate 117. This is to allow the receiver module 30 to catch up, in cycles, and therefore in iterations, to the status of the LFSRs 53, 54 in the fob 16.

Anytime that one of the buttons 12-15 on the fob is depressed, the fob will undergo one cycle, and the shift registers 53, 54 will undergo one or two, or two or three iterations, respectively. The pressings of the buttons 12-15 may occur simply by being crushed in a purse, children playing with the fob, or otherwise. Since each fob keeps its own LFSR generated numbers, and the receiver module 30 likewise maintains separate LFSR generated numbers for each fob, each fob will generally be able to track with the receiver module except for the inadvertent pressings of the switches 12-15. Whenever the switches 12-15 have caused a cycle that is not responded to by the receiver module 30, the first time the switches are pressed and the receiver module does respond, the content of the LFSRs 115, 116 will not compare with the corresponding bits of the 39 bit shift register 108. However, provision is made in accordance with the invention to allow the receiver module 30 to initiate additional cycles, and

the additional one or two iterations for the LFSR 115 and two or three iterations for the LFSR 116, so as to catch up to the fob. To this end, an 8 bit counter 143 allows the receiver module 30 to try to catch up to the fob in question by repeating as many as 256 cycles, automatically. In a normal case, the receiver module 30 will catch up to the fob in only a few cycles. But if the receiver module is more than 256 cycles behind, as may occur by repetitive pressings of one of the switches 12-15 in a suitcase or handbag, then the LFSRs 115, 116 will not match up with the 39 bit shift register 108. The receiver module 30 is non-responsive to incoming signals while it is attempting authentication of a previous signal; the 256 attempts to catch up will transpire in only a half second or less; thus, authentication will not be hampered by repetitive pressing of the unlock button 13 due to impatience. Eventually, the operator will understand that the receiver module is out of synchronization (cryptographic synchronization), and will press two buttons at one time (such as lock and unlock), or some other combination that will be recognized in the fob as a command to effect cryptographic resynchronization between the receiver module 30 and the fob 16, as well as to reinitialize following a loss of battery power (dead or changed), which allows the RAM data to disintegrate.

A recognized command to synchronize ("resynch command", hereinafter) in the command generator 77 (Fig. 1) will produce the panic/resynch bit on the line 81. The resynchronization process in accordance with the present invention includes returning to the beginning; that is, returning to the use of the secret initial values and starting all over again. As described hereinbefore, the resynch command is used to

initialize the units in the first place, and when they become out-of-synch, they are in a sense reinitialized just as when they are new. To that end, the panic/resynch bit on the line 81 will cause the two initial secret values to be loaded from the PROM 19 to the LFSRs 53, 54 and the two initial secret feedback masks to be made available to the exclusive ORs 55, 56, and the 39 bit secret feedback mask to be made available to the exclusive OR 70. The panic/resynch command on the line 145 causes the clocking gates 64 to cause 20 iterations, respectively, of the LFSRs 53, 54. The purpose is that, utilizing as many iterations as there are bits in the word, causes the maximal mix of the feedback, regardless of what the mask is, to assure complete bit-wise convolution. In this case, however, two additional changes from normal occur: the eight low order bit positions of the shift register 53 are provided with a truly random number on a trunk of 8 lines 146 from an 8 bit counter 145 which is allowed to respond to the clock 65 in a manner related to pressing of the buttons 12-15, as described with respect to Fig. 3 hereinafter. Since it is impossible for persons to depress buttons carefully enough to achieve other than a random number at computer clocking frequencies (500 KHz or more), the likelihood of this number being exactly the same in successive resynch processes is extremely small. After twenty iterations of feedback shifting, with the low order 8 bit positions of the LFSR 53 comprising those from the counter 145, outputs of the 20 bit LFSR 53 and the 19 bit LFSR 54 are provided to the 39 bit shift register 69. The shift register 69 thereafter undergoes 39 feedback shifting iterations, of the type described hereinbefore, to produce the 39 bit encrypted word in the 64 bit word 25. As before, the 16 bit ID for the

fob is provided to the word 25, along with a panic/resynch bit (described hereinbefore) to indicate that this is a panic or resynch request, and the error correction code is computed and the code bits added to the word 25 as described hereinbefore. The 20 bit LFSR and 19 bit LFSR results, after 20 iterations, form the pseudo random starting words to be used in authenticating future transmissions.

In the receiver module 30, the first two steps are the same as in a normal command. Error correction is provided if possible, and if the word is correct, the first OK signal appears on the line 94. Then, the four possible IDs are compared with the incoming ID in the word 38, and if there is a match, the second OK signal appears on the line 101 and the signals on the trunk of four lines 102 tell the EE PROM 40 which fob is being worked with and therefore which of the sets of two secret initial values and three secret feedback masks should be utilized. The appropriate secret initial values and three feedback masks are loaded into the LFSRs 115, 116, and the exclusive ORs 113, 114 and 109. The content of the 39 bit shift register 108 is reconstructed by 39 reverse iterations, as described hereinbefore, so as to recover the word in the 39 bit shift register 69. However, since the output of the 20 bit LFSR 53 does not reflect 20 shift iterations of only the secret initial value that was placed therein, but rather represents 20 iterations of 12 high ordered bits of the secret initial value and 8 random low ordered bits, comparisons with the high order bits of the 39 bit shift register 108 cannot be made in the receiver module 30. Instead, the 20 bit LFSR value must be recovered in the same way that the 39 bit shift register value is recovered. That is, a reverse linear feedback shift register operation,

utilizing the exclusive OR mask with the least significant bit, is achieved in a 20 bit LFSR 151 (Fig. 2) in association with a 20 bit exclusive OR 152. This restores the unscrambled number in which the 12 high order bit positions of the LFSR 151 should be the same as the 12 high order bit positions of the secret initial value in the 20 bit LFSR 115, and the low order 8 bit positions of the 20 bit LFSR 151 are some random number (produced by the counter 145).

The next step in the resynch process is to compare the high order 12 bit positions of the reconstituted word in the LFSR 151 with the 12 bits of the secret initial value of the 20 bit LFSR 115. Thus, the 12 bits on the trunk of 12 lines 160 are compared with the 12 bits on the trunk of 12 lines 161, which are created solely in response to the initial secret value. And, the 14 bits on the trunk of lines 127 are compared with the 14 bits on the trunk of lines 128; these should also compare because the 19 bit LFSR 116 has been passed through 20 iterations in response to its secret initial value so it should match the result in the 19 bit LFSR 54, the 14 high order bit positions of which have been reconstituted in the 14 bits of the 39 bit shift register 108 to which the trunk of lines 128 respond.

If both the 12 bit and 14 bit comparisons are successful, a determination is made whether the panic/resynch bit, provided on a line 150 from the 64 bit word 38, had been caused by a panic command or by a resynch command. If a resynch or panic command was sent, the resynch or panic command would have been exclusively ORed into the five low order bits of the 19 bit shift register 54, as described with respect to other commands hereinbefore. Therefore, the command will be extracted by the five bit exclusive OR 131 and

provided over the trunk of lines 137 to the command register 138. Since performing the panic command cannot breach vehicle security, it is used as the default command; if the resynch command is not present on the line 140, then the panic/resynch command on the line 150 may be deemed to be a panic command, if desired, even if not decoded. Up to this point, the panic command and the resynch command are identical.

The next step in the resynch process is to compare the 8 bit random number in the low ordered bit positions of the 20 bit shift register 151 with the last four prior low order 8 bit random numbers received during resynchronizing. In the present invention, the random number is compared with the last four such random numbers previously received by providing the 8 low order bit positions of the 20 bit LFSR 151 on a trunk of 8 lines 153 to 8 bits of the compare circuit 124 which are also responsive to a trunk of 8 lines 154 from a first in, first out stack 155 (actually embodied in the EE PROM 40), which keeps track of the last four 8 bit random numbers received during resynchronization operations. If, during resynchronization, the 8 central bits of the compare circuit 124 compare with any of the four 8 bit words in the first in, first out stack (FIFO) 155, the operation is a failure, and the receiver module 30 reverts to a half second wait period before it will react to the next command (as described hereinafter) and the matched word goes to the head of the stack and remaining words in the FIFO are adjusted accordingly. On the other hand, if the 8 bit word on the trunk of lines 153 does not compare with any of the bits in the stack 155, the comparison is a success and the operation can proceed; additionally, the 8 bit word on the trunk of 8 lines 153 is applied over the trunk of

lines 154 to the FIFO stack 155, for comparison with subsequent random 8 bit words during subsequent resynchronization operations. In such a case, the new word goes in the FIFO and the oldest word is dropped out of the FIFO.

Assuming that there is no match of the 8 bit random word, the resynchronization operation is complete. When the resynchronization is commanded, after successful comparisons of the 12 high order bits and the 14 bits as described hereinbefore and no comparison with the FIFO, the values established in the shift registers 53, 54, 115 and 116 are left as they are, for use in authenticating the next normal command cycle.

The panic command is the same as has been described with respect to the resynch command, except that, if the command register 138 produces the panic command signal on the line 140, the lights and the horn 44 (or other alarm) are operated, and, all of the LFSRs 53, 54, 115, 116 are then restored to whatever setting they had immediately before sending and receiving the panic command. The panic command operates differently from lock, unlock and trunk release commands, so that there will be response, even with total missynchronization between the fob and the receiver module. In the case of the panic command, starting over with the secret initial values ensures that authentication (to avoid nuisance responses) will be successful on the first try. Therefore, the panic command in the fob (Fig. 1) causes the LFSRs to be loaded with the initial values in the PROM 19, rather than the shift register values which had been achieved to date through iterations in the RAM 20, and the panic/resynch process just described is performed to

ensure that there will be authentication to execute the panic command.

The foregoing description is given as if it were hardware, and indeed the invention may be implemented in hardware along the lines described hereinbefore. However, the invention has been implemented in suitably programmed microprocessors, which are deemed most suitable. In the flowcharts described hereinafter, exemplary software routines are illustrative of the processing of the invention, but not necessarily of the individual steps of the program in any given embodiment of the invention.

The process of encryption in the fob is illustrated in Fig. 3. It is assumed that the fob comprises a microprocessor, such as a 68HC11, which has a stop mode in which the clock is stopped, the power consumption is negligible, and the only thing the processor can do is to respond to an external interrupt to get started again. In such a processor, application of battery power would cause the program to be reached through a power up entry point 170 and the processor would immediately stop at a step 171 where the only function is to perform a test 172 to determine whether any of the buttons 12-15 have been pressed, or not. So long as no button is pressed, the processor waits in a low power stop mode, in the loop 171, 172. As soon as a key is pressed, an affirmative result of test 172 reaches steps 173 in which a switch word in RAM 20 is ORed with the one of the switches which was pressed. Generally, two switches cannot be pressed within a few computer clocks of each other, so the first one will be sensed. As described hereinafter, if a second one is pressed within about 1/2 to one second, it will be treated as paired-up with the first; if the two are correct (e.g., lock and

unlock) a resynch command is declared. A switch interrupt, selectively enabled during normal command cycles only, allows sensing the second switch of a resynch.

In Fig. 3, the steps 173 also enable the switch interrupt and start the random counter. Then, a decode command subroutine 179 is performed and a test 180 determines if the command is either panic or resynch. If so, an affirmative result of test 180 reaches a series of steps 181 in which the shift register contents are saved in buffers, the panic/resynch (P/R) bit on line 81 is set to 1; a working register, herein referred to as a "shift register" (SR), is set with the random counter in its low order 8 bit positions and with the higher order bit positions equal to a 12 bit secret initial value for the 20 bit LFSR, the mask associated with the SR is set equal to the 20 bit secret feedback mask from the PROM 19, and a cycle counter C is set to 20 iterations. Then a bitwise linear feedback shift register iteration subroutine 182 is performed in which each bit is shifted to the next higher order position, with or without inversion, dependent upon the secret mask and/or whether the low order bit position has a 1, as described hereinbefore. Then the C counter is decremented in a step 183 and a test 184 determines if a complete, 20 iteration LFSR cycle has yet occurred. If not, another iteration is performed by the subroutine 182 and the C counter is decremented again. After 20 iterations, an affirmative result of the test 184 reaches a step 185 where the 20 bit shift register storage location in RAM 20 is set equal to the content of the working shift register. The steps and test 182-185 comprise an LFSR cycle 186.

Then the 19 bit shift register 54 is prepared in a series of steps 190 in which the content of the shift register is set equal to the content of the 19 bit secret initial value in the PROM 19, the mask associated with the shift register set equal to the 19 bit secret feedback mask in the PROM 19, and the C counter is set equal to 19. Then an LFSR cycle subroutine 191 (similar to the subroutine 186) is performed. Then the 39 bit shift register 69 is prepared for its LFSR cycle in a series of steps 192. Specifically, the 5 low order bit positions are the exclusive OR of the command with the 5 low stages of the 19 bit shift register 54; the high 14 bits of the 19 bit shift register 54 are placed directly in the 39 bit shift register; and the highest order 20 bit positions are set equal to the 20 bit positions of the 20 bit shift register 53. The mask is set equal to the secret feedback mask for the 39 bit shift register, found in the PROM 19, and the C counter is set to 39. Then, an LFSR cycle subroutine 193 is performed, this time with 39 iterations, and the result restored in the 39 bit shift register embodied in the scratch pad memory 20. In a routine 194, the 16 bits of the fob ID from the PROM 19, the 39 encrypted bits now in the 39 bit shift register, the P/R bit, and the download bit are all transmitted serially while the calculation for error correcting code bits is performed. These are calculated and transmitted, to complete the process of a panic or resynch command transmission. Whether it be a panic or a resynch is determined by the status of the five command bits. If a resynch was performed, the new values of the 20 bit shift register and 19 bit shift register will be retained as the pseudo random starting words to be used for future authentication of

transmissions to the receiving module. But if this is a panic command, the new values are only used to ensure synchronized response, one time, and a test 195 causes the previous values of the 20 bit and 19 bit shift registers to be restored from the buffers in a step 196.

Assuming that a normal command has been given, a negative result of the test 180 reaches a series of steps 199 in which the working shift register is set equal to the 20 bit shift register in the RAM 20 (not the secret initial value), so as to take advantage of the pseudo random number generated by all of the previous iterations. The mask for the shift register is set equal to the 20 bit shift register secret feedback mask from the PROM 19, and a set of random bits (which determines how many iterations are to be performed, similar to the gates 64 of Fig. 1) is set equal to whatever random bits have been selected to be used to control the iterations for the 20 bit shift register. This might, for instance, be the third and the ninth bit of the 20 bit shift register, or in a general case, can be anything else that is deterministically computable but difficult to predict. Then a pair of tests 200 determines what the random bits are: if both are a 1, a step 201 sets the C counter to 1; if both are a 0, a test 202 sets the C counter to 2. But if they are different, a step 201a sets the C counter to 3. Then, either a 1 iteration, 2 iteration or 3 iteration LFSR cycle subroutine 203 is performed. This aspect of the invention may be used in prior art systems, such as in the Hill et al patent. Next, the same sort of operation is accomplished with the 19 bit shift register; a series of steps 204 set the working shift register equal to the content of the 19 bit shift register in the ram

20, the mask for the working shift register is set equal to the secret feedback mask for the 19 bit shift register in the PROM 19, and the random bit is set equal to whatever bit has been chosen to be random for the 19 bit shift register. Then a test 205 determines if the random bit is 1, or not. If it is, a step 206 sets the C counter to 3, and otherwise a step 207 sets the C counter to 2. This provides four iterations (201, 206; 202, 207) whether the random bit is 1 or 0; but it may be set in other ways, if desired. Then a 2 or 3 iteration LFSR cycle subroutine 191 is performed. Then the series of steps 192 set things up so as to form the 39 bit encrypted word, a 39 iteration LFSR cycle subroutine 193 is performed so as to produce the 39 bit encrypted word, and the subroutine 194 transmits all the bits together with a calculated error correction code. Thus, the differences between encrypting and transmitting normal commands and the panic/resynch command are the setting of the P/R bit, the use of the random counter 145, the use of the secret initial values and the particular code which is exclusive ORed into the 19 bit shift register 54.

When a command word has been transmitted by the subroutine 194, and if a panic command, the shift registers have been restored from the buffer, the program advances to a one-half second wait in a step 208. This is to ensure that successive button pressings which are independent of each other will occur no closer than one-half second apart. During the time from when the computer was awakened by a command interrupt (at test 172 until the end of the one-half second waiting period at step 208), a switch interrupt might have occurred as a result of a second pressing of one of the switches 12-15. As described hereinbefore, this is most likely the case of an

attempt to press two switches at once (such as lock and unlock) to thereby cause a resynch. Whenever the switch interrupt is enabled, closing of one of the switches 12-15 will reach the interrupt subroutine of Fig. 4 through an entry point 209. In a series of steps 210, the particular switch which caused the present interrupt is remembered by being ORed into the switch word within the RAM 20; and since this may be a request for resynchronization, an internal resynch flag is set. The random counter is stopped, to provide the random number which is used in resynchronization, and then whatever part of the program of Fig. 3 was in process when the interrupt was sensed is returned to; this return may be to any of the functional steps ahead of the waiting step 208, or may be within the waiting step 208. Of course, if the waiting step 208 is interrupted, it will in fact turn out to be more than one-half second when the counting therefor is completed; this is irrelevant.

In Fig. 3, after the waiting period is over, a test 211 determines if the resynch flag has been set. If it has, an affirmative result of test 211 reaches a step 212 which resets the resynch flag, and then the program advances to the decode command subroutine 179. If the first switch which was pressed, turning on the computer, was either lock or unlock, and the second switch which was pressed, causing the switch interrupt, was either unlock or lock, respectively, then the decode command subroutine will in fact decode a resynch command, to cause a resynch operation of the type described hereinbefore. If not, any other two-key series may be decoded into a lock command for security, or into a panic command since the panic command will not affect security, or it could cause reversion to the one-half second waiting period, at

step 208, or otherwise as suits any particular implementation of the invention. Of course, if two switches which can cause a resynch command are pressed essentially simultaneously, the decode command subroutine will decode a resynch command without the aid of Fig. 4 and the resynch flag. If switches are repetitively pressed at less than half-second intervals, the switch word will either contain gibberish or will simply repeat the resynch command.

After waiting one-half second at the step 208, if the resynch flag has not been set, a negative result of test 211 will cause a pair of steps 213 to return the switch word to all zeros and to disable the switch interrupt, so that all future operation of the switches can only turn on the computer from its stop condition, at test 172. It should be noted that the command interrupt and the switch interrupt respond to the same thing: the operation of any of the switches; the difference is the microprocessor's response to them, as is well known in the art.

It is assumed that the decryption of Fig. 5 is carried out in a microprocessor of the same general type as is used in the fob. When connected to a battery, the routine is entered through a power up transfer point 214 and the processor immediately goes into a stop mode at a step 215, where the clock is off and the only function is to respond to a receiver interrupt at a test 216. In between usages, the processor in the automobile will remain in the stop mode, in the loop 215, 216. When an incoming message is sensed, an affirmative result of the test 216 will reach a subroutine 217 which handles receiving all 64 bits of the word transmitted from a fob, calculating the error correcting code, and fixing any single error which can be fixed. Then a test 218 determines if the

error correction code indicates correct data. If it does not, a negative result of test 218 reaches a transfer point 219 and then a step 220 where the program just waits for half of a second. The purpose of this is to severely hamper any attempts to break the code through repetitive application of numbers, with or without calculated likely candidates. After waiting one-half second, the processor returns to the stop mode in the loop 215, 216.

If the incoming word is OK, an affirmative result of test 218 reaches a step 221 where a working number, n, is set equal to 4 (or to such other number as the number of fobs which can be associated with the automobile). Then, a subroutine 223 compares all the bits of the incoming ID number to all the bits of the ID number for fob 4. If they are not equal, a negative result of a test 224 will reach a step 225 where n is decremented and a test 226 determines if all of the fobs have been checked or not. If they have, that means a signal has been received from a fob of another automobile by accident, or from some other unauthorized source. Therefore, an affirmative result of test 226 is taken to be a failure, and the wait step 220 is reached through the wait transfer point 219. Otherwise, the ID of another fob is checked in the subroutine 223. Assuming that the ID number matches for one of the fobs, another working number, N, is set equal to n so as to identify the words in the PROM and RAM needed for decryption, in a step 227. Since the received word may relate to a fob other than fob N, but having the same ID number, the shift register values for fob N are saved in a buffer, in a pair of steps 228, so they may be restored if authentication fails. This is also necessary since if a panic operation has been commanded, the iterations

of the shift registers continue in a normal fashion, after performing the panic command. Then a test 229 determines if the P/R bit was present in the incoming word, or not. If it is present in the incoming word, the panic/resynch decrypt routine of Fig. 6 is reached through a transfer point 230.

In Fig. 6, the first steps 232 set a main working shift register (SR) and its mask equal to the secret initial value and the secret feedback mask for the 19 bit LFSR, respectively, from the PROM 40 for the selected fob N, and a C counter is set equal to 20 so as to cause 20 iterations. Then, a 20 iteration LFSR cycle subroutine 233 is performed on the 19 bit shift register. It is assumed that the 39 bit encrypted word portion of the 64 bit received word 38 is stored immediately in a 39 bit shift register location within the RAM 41, which is where it now can be found. Then the 39 bit encrypted word, in the 64 bit word 38, and the 39 bit secret mask for the fob N are provided to the shift register and the C counter is set for 39 iterations, in step 234. Then, a bitwise reverse LFSR iteration subroutine 235 is performed which looks at the low order bit to determine whether the bits corresponding to the mask should be flipped before they are shifted to the next lower order position in the shift register to reconstitute the original word prior to encryption. After each iteration, the C counter is decremented in a step 236 and when all 39 iterations have been performed, an affirmative result of a test 237 will reach a step 238 wherein the content of the working shift register is stored in the 39 bit shift register in RAM 41. The steps and tests 235-238 comprise a reverse cycle subroutine 239, which recovers the

initial unencrypted value of the concatenation in the 39 bit shift register 69 in the fob.

Bits 19-38 of the decrypted 39 bit shift register are now loaded into the working shift register in steps 242, the mask for the shift register is set equal to the 20 bit secret feedback mask for fob N from the EE PROM 40, and C is set equal to 20, and a 20 iteration, reverse LFSR cycle subroutine 243 is performed to recover the combined word (initial value plus random). In a subroutine 244, there is a bitwise comparison of the 39 bit shift register bits 27-38 with the twelve bits of the 20 bit secret initial value for fob N, from EE PROM 40, and of bits 5-18 of the 39 bit shift register with bits 5-18 of the 19 bit shift register for fob N, which are found in the RAM 41. This is equivalent to the comparison of the 12 high order bits on the trunk of lines 160 with those on the lines 161 and of the 14 bits on the line 128 with the 14 bits on the lines 129, in Fig. 2. If these are not equal, the decryption is unsuccessful, the attempted access is a failure, and a negative result of a test 246 will reach a test 247 to see if another fob could match the ID; if it might, Fig. 5 is reverted to through a transfer point 248, the shift registers for fob N are restored from the buffers in steps 249 (Fig. 5), and the process is repeated for another fob. If all fobs have had their ID's checked, the routine reaches a pair of steps 250 where the shift registers for fob N are returned to their former values, and the program enters the half second wait at step 220 through the transfer point 219.

If the comparison is successful, indicating partial authentication, an affirmative result of test 246 reaches a subroutine 252 where the value in the

command register 138 is set equal to the exclusive OR of the low order bits of the 39 bit shift register and the low order bits of the 19 bit shift register, both taken from the RAM 41. Then, a test 253 determines if the panic/resynch bit was established in response to a resynch command. If not, the panic command is performed by turning on the lights, horn, or other alarm of the automobile in a step 254. Then, the shift registers for fob N are restored in the steps 250 and the wait step 220 is reached through the transfer point 219.

If the command were a resynch, a positive result of test 253 reaches a subroutine 255 which compares bits 19-26 of the 39 bit shift register (the regenerated random number) to a queue of previously used random numbers in the first in, first out stack (FIFO) 155, in EEPROM 40. If the random word compares to any of the last four (or whatever size FIFO is chosen) random words in the queue which were used in resynchronization, the resynchronization is deemed to be unsuccessful, since it is assumed that there has been clandestine playback of a copied random word. In such a case, an affirmative result of a test 256 reaches a subroutine 257 that rearranges the FIFO stack by moving the random word from its position in the queue in the FIFO stack to the first position thereof, and adjusting the position of the other words in the queue, without losing any. And, since this is deemed to be an unsuccessful attempt to resynchronize the unit, the old values in the 19 bit and 20 bit shift registers for fob N are restored in the steps 250. Then, the wait step 220 is reached through the transfer point 219.

If the random word did not compare with any word in the FIFO, a complete authentication exists, and a

negative result of test 256 reaches a subroutine 258 which rearranges the FIFO for fob N simply by adding bits 19-26 of the 39 bit shift register to the first location in the queue and shifting all the other words downward therein, causing the oldest word to fall out. In this case, the resynch operation is successful, so the steps 250 are bypassed and the setting of the 19 bit shift register for fob N in RAM 20 is left as it was established by the resynchronization operation, at subroutine 233. The 20 bit LFSR word created in the subroutine 243 is placed in the RAM for fob N (step 259), for use en futuro.

An important aspect of the present invention is that resynchronization occurs only after: 20 iterations of the 19 and 20 bit shift registers from their secret initial values and the random number; performing 39 iterations in the 39 bit shift register with those values, and the exclusive OR of the command; reverse iterations of the 39 bit encrypted word in the receiver module; reverse iteration of the 20 bits which include the secret initial value of the 20 bit shift register and the random counter; generation and successful comparison of the high order bits of the 19 bit shift register; and a failure of comparison of the random word with any of the last four random words used to resynchronize the system. This is quite secure.

Assuming that there is no P/R bit 81 in the 64 bit word which is received, a negative result of test 229 in Fig. 5 will reach the normal command decryption routine of Fig. 7 through a transfer point 261. In the normal command decryption process, the iterated 19 bit and 20 bit words in the RAM 20 are given 1-3 additional iterations and compared with the reverse-processed bits of the 39 bit encrypted word. As

described hereinbefore, since the fob may have its buttons pressed when the automobile cannot respond to it, they can become unsynchronized. Each time that a command is received in the receiver module 30, it is allowed 256 cycles to try to iterate to a correct pair of words that will match those which were transmitted to it. If it does so, then the command is responded to, and the iterated values are saved for authenticating the next command. If not, resynchronization is required, as described hereinbefore. In order to keep track of how many tries are made, the try counter 143 is set to its maximum count in a first one of a series of steps 262. Then, the working shift register (SR) is set equal to the 39 bit shift register in the RAM 20, which contains the 39 bit encrypted word. The mask for the shift register is set equal to the 39 bit secret feedback mask in the EE PROM 40, the C counter is set equal to 39, and a reverse LFSR cycle subroutine 267 is performed. Then a series of steps 271 cause the contents of the 20 bit shift register for fob N to be loaded from the RAM 41 into the working shift register (SR), the mask for the shift register is set equal to the secret feedback mask for fob N in the EE PROM 40 and a random bit is set equal to whatever bit or bits have been chosen for the 20 bit shift register (as described with respect to steps 199 in Fig. 3). Then the random bits are tested in a pair of steps 272, and if both are a 1, the C counter is set equal to 1 (step 273), if both are a 0, the C counter is set equal to 2 (step 274), and otherwise the C counter is set equal to 3 (step 274a). Then, an LFSR cycle subroutine 275 is performed. Then a series of steps and tests 276-279 prepare to run an LFSR cycle subroutine 280 for the 19 bit shift register in a similar fashion. Then

a subroutine 284 does bit-by-bit comparisons of the 39 bit shift register to the 20 bit shift register and to the high order bit positions of the 19 bit shift register, for fob N. If all the bits are equal, an affirmative result of a test 285 causes the particular command to be found by exclusive ORing the 5 low order bits in a subroutine 286 and the command is performed in a step 287, such as locking or unlocking the door, or releasing the trunk lid. But if the bits do not compare, a negative result of test 285 reaches a step 288 where the try counter is decremented and a test 289 to determine if 256 tries have been made yet. If not, another pair of iterations 275, 280 are performed and compared. This goes on until the receiver module 30 catches up with the fob 16 (if it can). If after 256 tries, a comparison has not been reached, an affirmative result of test 289 will reach a test 290 to see if all the fobs have been given consideration. If not, the restoration steps 249 (Fig. 5) are reached through the transfer point 248. Otherwise, the shift registers for fob N are restored to their starting values in steps 292 and the program reverts to the wait step 220 (Fig. 5) through the transfer point 219.

In Fig. 8, an alternative random iteration utilizes a pair of bits from the random counter to determine the number of iterations of the 20 bit shift register operation and one bit thereof to determine the number of iterations in the 19 bit shift register operation. In a set of steps 199a (in place of steps and tests 199-202a, Fig. 3), the cycle counter, C, has its two low ordered bit positions (0, 1) set equal to the selected bits (3 and 6 in this example) of the random counter, and it is incremented to ensure that a zero-valued pair of bits will not result in no iteration. Similarly, steps 204a replace steps and

test 204-207 to provide one or two iterations in the 19 bit operation. In Fig. 9, the random number is found at the head of the queue in the FIFO memory (40) for fob N. Steps 271a replaces steps and tests 271-274a and steps 276a replace steps and tests 276-279. In this embodiment, the number of iterations in each case is one higher (1-4; 1 or 2) than the value of the bit or bits (taken together as low ordered bits) selected from the random counter. However, fewer choices of the number of iterations may be used in response to random counter bits, as in tests 200 and 272, or by means of any other decode. Of course, more bits and higher numbers may be used.

In a system which uses only one or two fobs, or in which speed of response is not important, the present invention may be utilized without the use of ID numbers, simply by trying all of the sets of stored shift register words and masks; this would make the system subject to more nuisance iterations since the ID number would no longer screen out many transmissions from similar, unauthorized systems. And there must be some sort of a tag to correlate the generated pseudo random words with the correct feedback mask for subsequent iterations, as well as with the correct initial value. This may be achieved by arbitrary numbering of the various sets of initial values and masks and corresponding registers for storing the iterated words.

The various number of iterations used during synchronization may be the same (e.g., 20, or more) for both random words generated from initial values, or they may be different (e.g., 20 and 19, or more).

As used herein, the term "encryption" inherently defines a process capable of decryption. Thus, if the process of encryption employs a pseudo random number

generation operation of any type, that generation operation must be capable of being practiced in reverse, such as the reverse LFSR operation utilized herein. The encryption processes and operations used herein may be any of those described hereinbefore by way of example, and others. However, the LFSR pseudo random number generation operation is one of the easier ones to understand, and probably the simplest to implement in dedicated or quasi-dedicated hardware. That is, if the signal processing means of the invention includes dedicated shift registers and the like as may be implied in Figs. 1 and 2, the LFSR form of encryption may be preferred. Similarly, if the signal processing means of the present invention is implemented with microprocessors having suitable program routines as disclosed in Figs. 3-7 herein, the advantage of LFSR operations over other forms of encryption may be less distinct. The term "encryption" is, therefore, used herein in its broadest sense, so long as the word which becomes encrypted can be recovered through decryption.

In the foregoing embodiments, secret initial values and secret feedback masks for generating maximal length sequences of pseudo random numbers are essentially unique to each transmitter, but not necessarily totally unique. By this it is meant that the sets of two secret initial values and three secret feedback masks, taken together, are capable of distinguishing more vehicles than are made in a lifetime; however, the method of assignment, or human error or design, could result in some few with the same set of numbers. Within this definition, however, it is quite possible that several transmitters will have the same 19 bit secret initial value and/or the same 20 bit secret mask, or other similar

combinations. In contrast, the use of a 16 bit identification number means that there will be several fobs each year possibly having the same identification number (but not the same secret values) so that over a course of time, there could be many fobs (such as about 5,000 fobs in the USA over a 10-year period for each possible identification number). Within statistical probability, it is possible that as many as one out of 10,000 automobiles having four fobs each may have two fobs with the same identification number. It is an important aspect of the present invention, that even if one fob having a correct identification number is not authenticated within 256 tries (or whatever is used), it will nonetheless try any other fob that may have the same ID number. If trying three or four fobs results in too much delay, the last tried fob is likely to be reported as faulty, and a new fob issued, with very minuscule likelihood that the new fob would have the same identification number as the remaining fobs in the set.

Although the present invention employs only lock-related commands, the panic alarm command, and synchronization command, it should be understood that the invention can be utilized to authenticate conveyance of any information in the form of bits similar to the command bits herein. Thus, in its broadest sense, the term "lock-related command" means the conveying of other than a synchronization command, the conveyance of which is to be authenticated; this may then be thought of as a command apart from the internal functioning of the system itself.

In the present embodiment, each encryption (such as the 19 bit, the 20 bit and the 39 bit shift register encryptions) are the same (linear feedback shift register pseudo random number generation).

However, they need not all be the same, and in fact can be different algorithms to further confuse any attempted analysis. Similarly, the number of bits in the shift registers can be whatever is desired. In all cases, the greater number of bits, the harder to break the code by analysis. The numbers used herein provide a safe system, but greater or fewer numbers of bits may be chosen in practicing the invention if desired. Similarly, the words may be considered to be single words, or double words in the sense that the 19 bit shift register produces one portion of 14 bits which is used for cryptographic authentication, and another portion which carries the command but is not used in cryptographic comparison for authentication. These portions can be considered to be two different words except for the fact that in the embodiment herein they are generated in the same process. Of course, separate processes could be used, or two processes of a different split of numbers of bits could be used to encrypt and iterate the encryption of the word in which the command bits are found. For instance, in place of a single 19 bit shift register operation, a 10 bit shift register operation and a 9 bit shift register operation could be utilized, the results thereof concatenated, and five command bits exclusive ORed into a portion of one of them, before being used in the final encryption.

The invention may use more than two concatenated words in the final encryption, such as an additional word or such as having three words, each slightly smaller than the two words used in the final 39 bit encryption herein.

The invention is disclosed as being employed in system in which the transmitter transmits to the receiver, but the receiver does not transmit back to

the transmitter. However, the precepts of the present invention can be used singularly, or in combinations in systems which, for one purpose or another, employ bilateral communications between the two units. Aspects of the invention may then be used to authenticate transmissions in both directions, or only in one direction, as is necessary. Although the invention is disclosed herein, and may find its greatest utilization, in a remote authentication system, it may as well be used in a system in which the transmitter is connected by a conductor to the receiver, so as to provide secure operation between the two. Similarly, aspects of the invention may be utilized in contact embodiments, such as in electronic keys, so as to provide insurance against the temptation of insiders to perform a clandestine system breach, as well as protecting against outsiders compromising the system.

In the disclosed embodiments, the 20 bit shift register operation may employ one through five iterations, dependent upon the random occurrence of a pair of bits within the register, or a pair of random bits. Similarly, the 19 bit shift register operation may employ one through three iterations, depending upon one of its bits or a random bit. In the general case, these numbers may as well be reversed, they may be the same, they may depend on each other or on any pseudo random events that can be duplicated in the receiver, and/or either of them may be greater, provided that time constraints do not prohibit a greater number of iterations, and that every possible condition results in at least one iteration. It is to be noted that the small number of iterations would not be important but for the fact that in any given transmission, the receiver is allowed up to 256

2159360

Claims

1. A method of cryptographic authentication of transmissions from a transmitting unit to a receiving module,

comprising, in said transmitting unit:

separately generating a plurality of pseudorandom numbers;

concatenating said numbers to form a combined word;

performing an encryption operation on said combined word; and

transmitting a command word including a key portion derived from the result of said encryption operation; and

comprising, in said receiving module:

receiving said command word;

performing a decryption operation on the key portion of said command word to recover said combined word;

providing at least one number; and

providing an authentication signal only if at least a portion of said at least one number is identical to a corresponding portion of said recovered combined word.

2. A method according to claim 1 wherein:

said step of providing at least one number comprises separately generating a second plurality of pseudorandom numbers; and

said step of providing an authentication signal comprises providing said authentication signal only if at least a portion of each of said second plurality of pseudorandom numbers is identical to a corresponding portion of said recovered combined word.

3. A method according to claim 2 wherein said generating steps each comprise generating a pair of numbers.

4. A method according to claim 1 wherein said generating step comprises generating a pair of numbers.

5. A method according to claim 1 wherein said step of performing an encryption operation comprises performing a linear encryption operation.

6. A method according to claim 1 wherein said step of performing an encryption operation comprises performing a feedback shift register operation.

7. A method according to claim 6 wherein said step of performing an encryption operation comprises performing a linear feedback shift register operation employing a secret feedback mask and said step of performing a decryption operation comprises performing a reverse linear feedback shift register operation

employing the same secret feedback mask as in said encryption operation.

8. A method according to claim 7 wherein said linear feedback shift register operation comprises a number of iterations on the order of the degree of said combined word or more.

9. A method of cryptographically authenticating a transmission from a transmitting unit to a receiving module, comprising:

providing a starting number in said transmitting unit and providing said starting number in said receiving module;

in said transmitting unit:

providing an iteration control signal which changes in a pseudorandom manner in response to successive transmissions from said transmitting unit;

performing a variable number of iterations of an iterative encryption operation on said starting number, said variable number determined by said iteration control signal;

transmitting a command word derived at least in part from the result of said encryption operation; and

in said receiving module:

receiving said command word;

recovering the result of said encryption operation from said received command word;

providing a second iteration control signal which changes, in the same pseudorandom manner as said first iteration control signal, in response to successive receptions of command words by said receiving module;

performing a variable number of iterations of said iterative encryption operation on said starting number, said variable number determined by said second iteration control signal;

comparing at least a portion of the result of said encryption operation performed in said receiving module with a corresponding portion of said recovered result; and

providing an authentication signal only if said portion of said encryption operation performed in said receiving module is identical to said corresponding portion of said recovered result.

10. A method according to claim 9 wherein said step of performing an iterative encryption operation comprises performing a linear iterative encryption operation.

11. A method according to claim 9 wherein said step of performing an iterative encryption operation comprises performing a feedback shift register operation.

12. A method according to claim 9 wherein said step of performing an iterative encryption operation comprises performing a linear feedback shift register operation

employing the same secret feedback mask in said transmitting unit as in said receiving module.

13. A method according to claim 9 wherein:

the same secret initial value is provided in said transmitting unit and in said receiving module; and

said starting number is provided by performing said iterative encryption operation on a word derived at least in part from said secret initial value a number of iterations on the order of the degree of said word, or more.

14. A method according to claim 9 wherein said variable number of iterations is a fraction of the degree of said starting number.

15. A method according to claim 9, comprising:

providing a second starting number in said transmitting unit and providing said second starting number in said receiving module;

in said transmitting unit:

providing a third iteration control signal which changes in a pseudorandom fashion in response to successive transmissions from said transmitting unit;

performing a changeable number of iterations of an iterative encryption process on said third starting number, said changeable number determined by said third iteration control signal;

transmitting said command word derived at least in part from the result of said encryption process; and

in said receiving module:

recovering the result of said encryption process from said received command word;

providing a fourth iteration control signal which changes, in the same pseudorandom fashion as said third iteration control signal, in response to successive receptions of command words by said receiving module;

performing a changeable number of iterations of said iterative encryption process on said starting number, said changeable number determined by said fourth iteration control signal;

comparing at least a portion of the result of said encryption process performed in said receiving module with a corresponding portion of said recovered result of said encryption process; and

providing an authentication signal only if said portion of said encryption process performed in said receiving module is identical to said corresponding portion of said recovered result of said encryption process.

16. A method according to claim 15 wherein said variable number is different from said changeable number.

17. A method according to claim 15 wherein said pseudorandom manner is different from said pseudorandom fashion.

18. A method according to claim 15 wherein said iterative encryption operation is the same as said iterative encryption process.

19. A method of cryptographically authenticating transmissions from any of a plurality of remote command transmitting units to a command performing receiving module, comprising:

providing a set of numbers in each of said transmitting units, each set corresponding to one of said transmitting units, each set including at least one secret initial value, each set essentially unique to the corresponding unit;

providing in said receiving module, said set of numbers for each of said transmitters to which said receiving module is to respond;

transmitting a command word from one of said transmitting units including a key portion derived at least in part from an encryption operation performed on said secret initial value; and

authenticating said command word received at said receiving module utilizing the numbers in a corresponding set.

20. A method according to claim 19 wherein each of said sets includes an identification number;

said transmitting step comprises transmitting said command word including said identification number; and

said authenticating step comprises performing a process to authenticate said received command word only in response to said command word containing an identification number which matches an identification number in one of the sets provided in said receiving module.

21. A method according to claim 20 wherein, in response to receipt of said command word, said receiving module performs an authentication process using successive ones of said sets which have an identification number that matches the identification number included in said received command word until either authentication occurs or all of said sets have been used.

22. A method according to claim 19 wherein, in response to receipt of said command word, said receiving module performs an authentication process on said key portion using successive ones of said sets until either authentication occurs or all of said sets have been used.

23. A method according to claim 19 wherein each set includes at least one corresponding secret feedback mask,

and said encryption operation comprises a feedback shift register pseudorandom number generation operation utilizing said secret feedback mask.

24. A method according to claim 23 wherein said shift register operation is linear.

25. A method of synchronized cryptographic authentication of transmissions from a remote command transmitting unit to a command performing receiving module selectively responsive thereto comprising:

transmitting a command word including a key portion derived from at least one encrypted number generated in said transmitting unit and indicative of a command;

receiving said command word and, in response thereto, comparing a number in said receiving module with a number decrypted from the key portion recovered from said command word, providing an authentication signal based at least in part on identity between said number in said receiving module and said number decrypted from the key portion recovered from said command word, selectively performing the command indicated thereby in response to said authentication signal; and

rendering said receiving module unresponsive, following receipt of one command word, to receipt of an additional command word for a period of time on the order of one-half second, or more.

26. The method according to claim 25 wherein said rendering step comprises providing a waiting period between the conclusion of any operation responsive to receipt of one of said command words and the enabling of said receiving module to be responsive to a subsequently received command word.

27. A method of selectively cryptographically authenticating transmissions, indicative of commands initiated by operating switches, from each of a plurality of transmitting units to a receiving module, comprising:

providing a set of numbers in each one of said transmitting units, each set corresponding to one of said transmitting units and identified by an identification number, each set including at least a pair of secret initial values;

providing in said receiving module the one of said sets corresponding to each of said transmitting units to which said receiving module is to respond;

in response to operation of said switches indicating a command other than a lock-related command in one of said transmitting units:

providing a command bit;

generating a random number;

concatenating said random number with a first one of said secret initial values so as to provide a combined word;

performing a first encryption operation on said combined word to provide a first number;

performing a second encryption operation on a second one of said secret initial values to provide a second number;

exclusive ORing a plurality of command bits indicative of said command with the corresponding bits of said second number to provide an altered word;

performing a third encryption operation on the concatenation of said first number with said altered word to provide an encrypted key word;

storing said first and second numbers as first and second pseudorandom numbers for future use in subsequent authentication;

transmitting a command word including said encrypted key word, said command bit, and said identification number;

in response to operation of said switches indicating a lock-related command in one of said transmitting units:

performing a fourth encryption operation on said first number to provide a new first pseudorandom number;

performing a fifth encryption operation on said second number to provide a new second pseudorandom number;

exclusive ORing a plurality of command bits indicative of said lock-related command with the corresponding bits of said new second pseudorandom number to provide a new altered word;

performing a sixth encryption operation on the concatenation of said new first pseudorandom number and said new altered word to provide a new encrypted key word;

storing said new first and second pseudorandom numbers for future use in subsequent authentication in place of said first and second pseudorandom numbers;

transmitting a command word including said new encrypted key word and said identification number;

in said receiver, selectively, in response to receipt of said command word including said command bit:

determining if said receiver has secret initial values related to the received identification number, and if not, terminating all response to said received word, but if so:

performing a first decryption operation on said key word portion of said received command word so as to recover said first number and said altered word;

performing, on said recovered first number, a second decryption operation so as to recover said combined word, comparing said first secret initial value to a corresponding portion of said recovered combined word and providing a first equal signal only in the event of identity therebetween;

performing a seventh encryption operation on said second secret initial value to provide said second number, comparing the non-command portion of said recovered altered word with the corresponding portion of

said second number and providing a second equal signal only in response to identity therebetween;

then, in response to the absence of either of said first and second equal signals, terminating all further response to said command word;

or otherwise, in response to the presence of said first and second equal signals, comparing the random number portion of said recovered combined word to a random number portion derived from a command word previously received from said transmitter and, in response to identity therebetween, terminating all further response to said command word, but otherwise, (a) storing said random number portion for future use in subsequent synchronization operations and (b) storing said second number and said recovered first number, as first and second pseudorandom numbers for future use in subsequent authentication operations;

in said receiver, selectively, in response to receipt of said command word not including said command bit:

determining if said receiver has secret initial values related to the received identification number, and if not, terminating all response to said received word, but if so:

performing a third decryption operation on the key word portion of said received command word, so as to recover said new first pseudorandom number and said new altered word;

performing an eighth encryption operation on said first pseudorandom number to provide a first new pseudorandom number, and comparing said first new pseudorandom number to said recovered new first pseudorandom number and providing a third equal signal in response to identity therebetween;

performing a ninth encryption operation on said second pseudorandom number to provide a second new pseudorandom number and comparing the non-command portion of said recovered new altered word to a corresponding portion of said second new pseudorandom number and providing a fourth equal signal only in response to identity therebetween;

then, in the absence of either of said third or fourth equal signals, terminating all further response to receipt of said command word, but in the presence of both of said third and fourth equal signals, exclusive ORing the command portion of said recovered new altered word with the corresponding portion of said second new pseudorandom number, performing the command indicated by the result thereof, and storing said first new pseudorandom number and said second new pseudorandom number for future use in subsequent authentication operations.

28. A method according to claim 27 wherein, in response to the presence of said first and second equal signals, the command portion of said recovered new

altered word is exclusive ORed with the corresponding portion of said second new pseudorandom number and said steps (a) and (b), are performed in response to the result of said exclusive OR operation indicating said command is a synchronization command.

29. A method according to claim 27 wherein said encryption operations comprise linear feedback shift register operations.

30. A method according to claim 27 wherein said first, second and third encryption operations employ the same algorithm.

31. A method according to claim 27 wherein said first and fourth encryption operations employ the same algorithm.

32. A method according to claim 27 wherein said second and fifth encryption operations employ the same algorithm.

33. A method according to claim 27 wherein said third and sixth encryption operations employ the same algorithm.

34. A cryptographically authenticated remote control system in which a command transmitting unit may

selectively cause a physical effect in a command receiving module rendered responsive thereto;

said transmitting unit comprising:

a source of signals for providing first and second seed signals indicative of respective secret pseudorandom number generator initial values and first, second and third mask signals indicative of respective secret feedback masks, each mask defining a respective feedback polynomial for linear feedback shift register pseudorandom number generation, said initial values and said polynomials being essentially unique to said transmitting unit;

command switches operable to indicate a physical effect which is to be caused by said receiving module; and

first signal processing means responsive to selected operation of said switches indicative of a synchronization command for providing a random signal indicative of a variable random number, for performing a first linear feedback shift register pseudorandom number generation operation, on a combined number consisting of the initial value defined by said first seed signal concatenated with the random number defined by said random signal, a given number of iterations on the order of the degree of said first polynomial, or more, using the mask defined by said first mask signal, said first polynomial having a degree on the order of the degree of said combined word, for performing a second linear

feedback shift register pseudorandom number generation operation, on a second word consisting of the initial value defined by said second seed signal, a fixed number of iterations on the order of the degree of said second polynomial, or more, using the mask defined by said second mask signal, said second polynomial having a degree on the order of the degree of said second initial value, for exclusive ORing a plurality of command bits indicative of said synchronization command with a corresponding plurality of bits of the result of said second generation operation to form an altered word, for storing, for future use in authenticating subsequent transmissions to said receiving module, first and second pseudorandom numbers respectively indicative of the results of said first and second generation operations, for performing a third linear feedback shift register pseudorandom number generation operation, on a word consisting of said first pseudorandom number concatenated with said altered word, a predetermined number of iterations on the order of the degree of said third polynomial, or more, using the mask defined by said third mask signal, said third polynomial having a degree on the order of the summation of the degrees of said first pseudorandom number and said altered word, and for transmitting, to said receiving module, a command word signal having the result of said third generation operation as a key portion and including a command bit indicative of said synchronization operation;

said first signal processing means responsive to selected operation of said switches indicative of a lock-related command for performing a fourth linear feedback shift register pseudorandom number generation operation, on said first pseudorandom number, a first determined number of iterations, using the mask defined by said first mask signal, to provide a new first pseudorandom number, for performing a fifth linear feedback shift register pseudorandom number generation operation, on said second pseudorandom number, a second determined number of iterations, using the mask defined by said second mask signal, to provide a new second pseudorandom number, for exclusive ORing a plurality of command bits indicative of said lock-related command with a corresponding plurality of bits of said new second pseudorandom number to form a new altered word, for performing a sixth linear feedback shift register pseudorandom number generation operation, on a word consisting of said new first pseudorandom number concatenated with said new altered word, said predetermined number of iterations, using the mask defined by said third mask signal, for storing said new first and second pseudorandom numbers for future use in authenticating subsequent transmissions to said receiving module, and for transmitting, to said receiving module, a command word signal having the result of said third generation operation as a key portion;

said receiving module comprising:

a signal source for providing third and fourth seed signals respectively indicative of said initial values and fourth, fifth and sixth mask signals respectively indicative of said masks; and

second signal processing means for receiving said command word signal and responsive to said command word including said command bit, for performing a first reverse linear feedback shift register pseudorandom number generation operation, on said key portion of said command word signal, said predetermined number of iterations, using the mask defined by said sixth mask signal, for performing a second reverse linear feedback shift register pseudorandom number generation operation, on a portion of the result of said first reverse generation operation corresponding to said combined word, said given number of iterations, using the mask defined by said fourth mask signal, for comparing said initial value defined by said third seed signal with an equivalent portion of the result of said second reverse generation operation and providing a first equal signal only if they are identical, for performing a seventh linear feedback shift register pseudorandom number generation operation on a word consisting of the initial value defined by said fourth seed signal, said fixed number of iterations, using the mask defined by said fifth mask signal, for comparing a portion of the result of said seventh generation operation, corresponding to

the unaltered portion of said altered word, with a corresponding portion of the result of said first reverse generation operation and providing a second equal signal only if they are identical, in response to said first and second equal signals, for storing, for subsequent use, the random number portion of the result of said second reverse operation and for comparing said random number portion with a similar random number portion, previously stored for subsequent use in response to prior performances of said second reverse operation, and for selectively storing third and fourth pseudorandom numbers respectively indicative of the result of said second reverse operation and said seventh generation operation, for future use in subsequent authentication of transmissions from said transmitting unit, only if said compared random portions are not equal;

said second signal processing means responsive to said command word signal not including said command bit for performing a third reverse linear feedback shift register pseudorandom number generation operation, on said key portion of said command word signal, said predetermined number of iterations, using the mask defined by said sixth mask signal, to recover said new first pseudorandom number and said new modified word, for performing an eighth linear feedback shift register pseudorandom number generation operation, on said third pseudorandom number, said first determined number of iterations, using the mask defined by said fourth mask

signal, to provide a third new pseudorandom number, for comparing said recovered new first pseudorandom number with said third new pseudorandom number and generating a third equal signal only if they are identical, for performing a ninth linear feedback shift register pseudorandom number generation operation, on said fourth pseudorandom number, said second determined number of iterations, using the mask defined by said fifth mask signal, to provide a fourth new pseudorandom number, for comparing the non-command portion of said recovered new altered word with a corresponding portion of said fourth new pseudorandom number and providing a fourth equal signal only if they are identical, and, in response to said first and second equal signals, for storing for future use in subsequent authentication of transmissions from said transmitting unit, new third and fourth pseudorandom numbers indicative of the results of said eighth and ninth generation operations, for exclusive ORing the command portion of said recovered new altered word with the corresponding portion of said fourth new pseudorandom number to recover said plurality of command bits and for performing said lock-related command.

35. A system according to claim 34 wherein the initial value indicated by said first seed signal is different from the initial yalue defined by said second seed signal.

36. A system according to claim 34 wherein said polynomials are all different from each other.

37. A system according to claim 34 wherein said fixed number is equal to said given number.

38. A system according to claim 34 wherein said first determined number is different from said second determined number.

39. A system according to claim 34 wherein said feedback polynomials are maximal length feedback polynomials.

40. A system according to claim 34 wherein said first and second determined numbers each vary as a function of a respective pseudorandom event, responsive to each transmission in said transmitting unit and responsive to each reception in said receiving module.

41. A system according to claim 40 wherein said first and second determined numbers are a fraction of said given number and said fixed number, respectively.

42. A method according to claim 9 wherein said iteration control signal changes in response to the value of a bit position of a changing number.

43. A method according to claim 9 wherein said iteration control signal changes in response to the value of a plurality of bit positions of a changing number.

44. A method according to claim 9 wherein said iteration control signal changes in response to the value of a bit position of said starting number.

45. A method according to claim 57 wherein said iteration control signal changes in response to the value of a plurality of bit positions of a random number.

46. A method of cryptographically authenticating a transmission from a transmitting unit to a receiving module, comprising:

providing a starting number in said transmitting unit and providing said starting number in said receiving module;

in said transmitting unit:

providing an iteration control signal which changes in a random manner in response to successive transmissions from said transmitting unit;

performing a variable number of iterations of an iterative encryption operation on said starting number, said variable number determined by said iteration control signal;

transmitting a command word derived at least in part from the result of said encryption operation; and

in said receiving module:

receiving said command word;

recovering the result of said encryption operation from said received command word;

providing a second iteration control signal which changes, in the same random manner as said first iteration control signal, in response to successive receptions of command words by said receiving module;

performing a variable number of iterations of said iterative encryption operation on said starting number, said variable number determined by said second iteration control signal;

comparing at least a portion of the result of said encryption operation performed in said receiving module with a corresponding portion of said recovered result; and

providing an authentication signal only if said portion of said encryption operation performed in said receiving module is identical to said corresponding portion of said recovered result.

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 9/32, E05B 49/00 — A3

(11) International Publication Number: WO 95/01685

(43) International Publication Date: 12 January 1995 (12.01.95)

(21) International Application Number: PCT/US94/07337

(22) International Filing Date: 29 June 1994 (29.06.94)

(30) Priority Data:
08/086,080 30 June 1993 (30.06.93) US

(71) Applicant: UNITED TECHNOLOGIES AUTOMOTIVE, INC. [US/US]; 5200 Auto Club Drive, Dearborn, MI 48126-9982 (US).

(72) Inventors: KOOPMAN, Philip, J., Jr.; 48 Willow Drive, Hebron, CT 06248 (US). FINN, Alan, M.; 26 Mohegan Lane, Amston, CT 06231 (US).

(74) Agent: SLENZAK, Laura, M.; United Technologies Automotive, Inc., Legal Staff - Patent, 5200 Auto Club Drive, Dearborn, MI 48126-9982 (US).

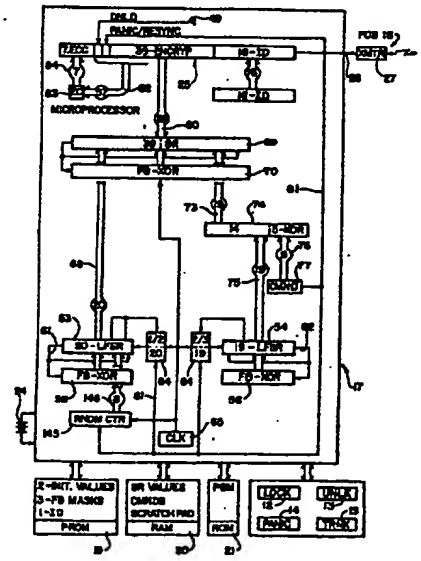
(81) Designated States: CA, JP, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(88) Date of publication of the international search report:
16 February 1995 (16.02.95)

2159360

(54) Title: METHOD FOR CRYPTOGRAPHIC AUTHENTICATION OF TRANSMITTED MESSAGES USING PSEUDORANDOM NUMBERS



(57) Abstract

An automobile door lock receiver module (30) and a plurality of keychain fob transmitter units (16) contain identification numbers, secret initial values, and secret feedback masks so as to authenticate encrypted messages from any of the assigned fobs, indicative of commands registered by closing switches on the fob. Each fob is synchronized with the receiving module by means of a truly random number concatenated with a secret initial value and encrypted, through a linear feedback shift register or other operations. A second secret initial value is encrypted and command bits are exclusive ORed into the low order bit positions; the two encrypted numbers are concatenated and encrypted to form a key word which is transmitted with the fob ID. Synchronization includes decrypting to recover the truly random number and the secret initial value concatenated therewith; the truly random number is compared with previously received random numbers in order to avoid copying of recently transmitted synchronization commands. Successive lock-related commands utilize the number encrypted from the truly random number and the second secret initial value as starting values, employing a pseudo random number of encryption iterations. A half-second delay between responses mitigates gaining access through numerical trials. An authenticated panic alarm command operates the headlights and horn of the vehicle but does not alter the synchronization.

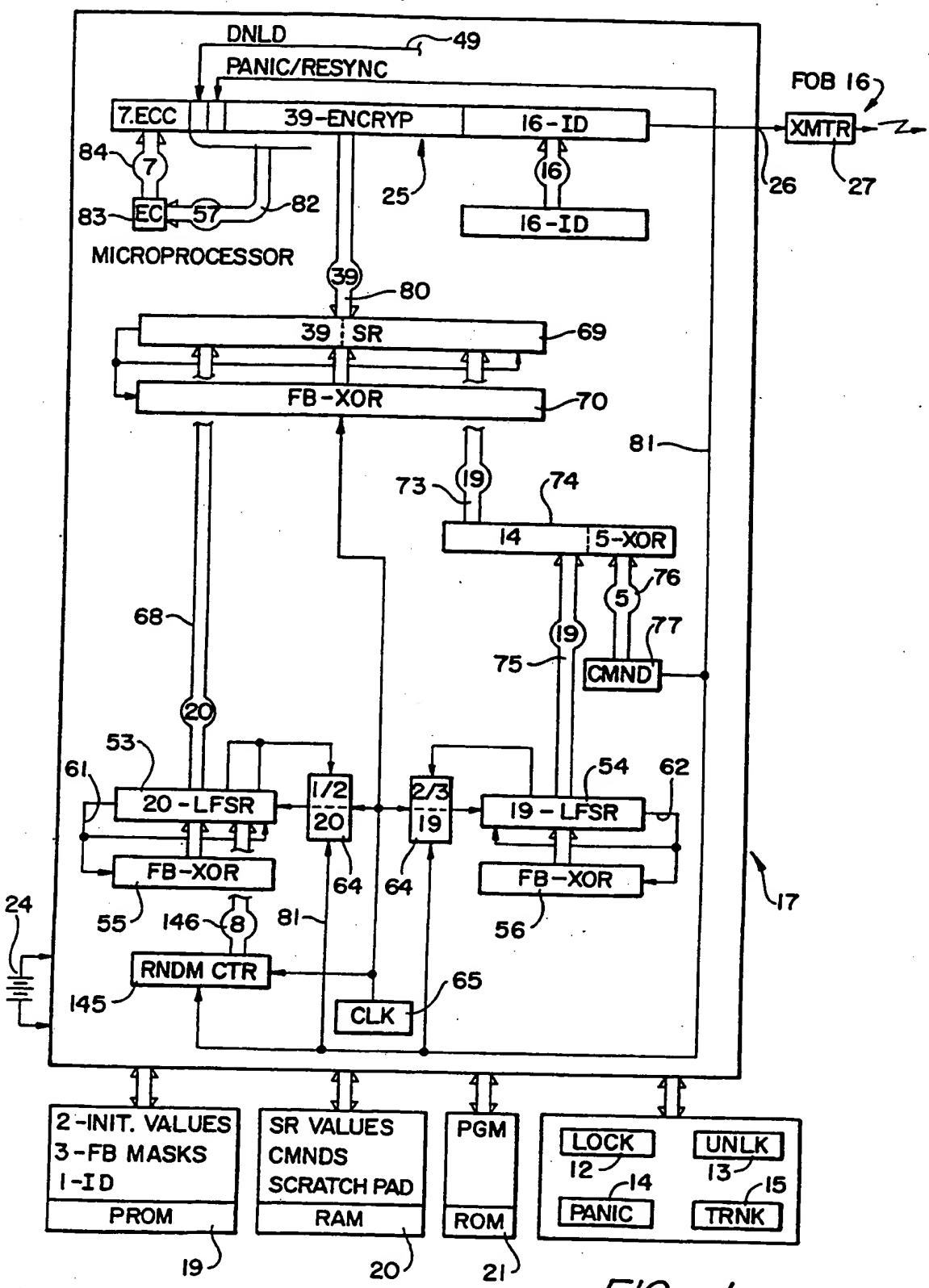

DNLD
49
PANIC/RESYNC
FOB 16
7.ECC
39-ENCRYP
16-ID
XMTR
84
7
16
25
26
27
83
EC
57
82
16-ID
MICROPROCESSOR
39
80
39 SR
69
FB-XOR
70
81
73
19
74
14
5-XOR
76
5
68
19
77
75
CMND
20
53
61
20-LFSR
1/2
20
2/3
19
19-LFSR
54
62
FB-XOR
64
64
FB-XOR
24
55
146
8
81
56
17
RNDM CTR
145
CLK
65
2-INIT. VALUES
3-FB MASKS
1-ID
PROM
19
SR VALUES
CMNDS
SCRATCH PAD
RAM
20
PGM
ROM
21
LOCK
12
UNLK
13
14
15
PANIC
TRNK

FIG. 1

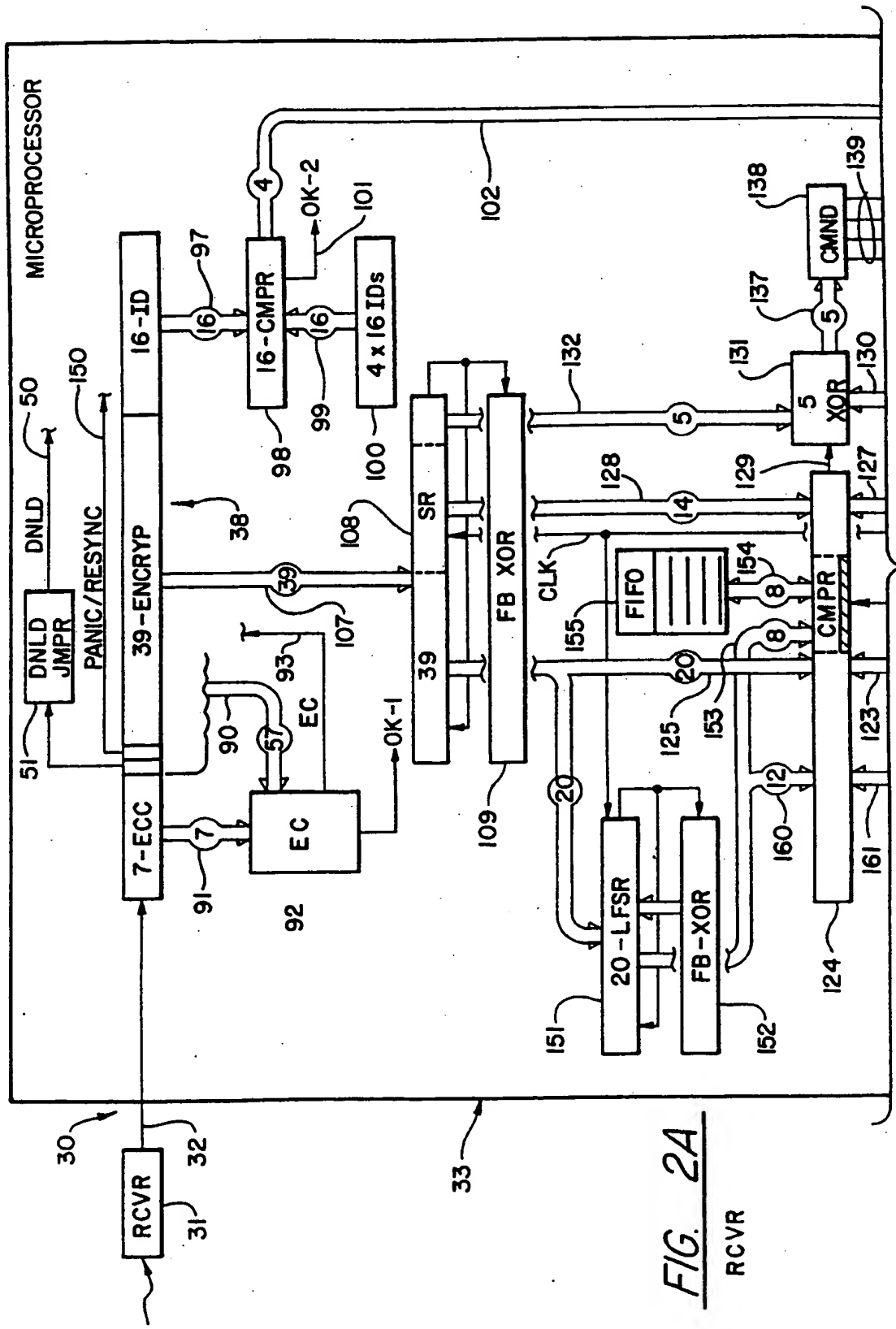

MICROPROCESSOR
16-ID
39-ENCRYP
7-ECC
DNLD JMPR
DNLD
PANIC/RESYNC
16-CMPR
4 x 16 IDs
OK-2
OK-1
EC
SR
39
FB XOR
FIFO
CMPR
CMND
5 XOR
20-LFSR
FB-XOR
RCVR
50
51
150
38
97
98
99
100
101
102
90
91
92
93
107
108
109
132
131
130
138
139
137
128
129
127
154
155
125
153
123
160
161
124
151
152
30
31
32
33

FIG. 2A

FIG. 2B

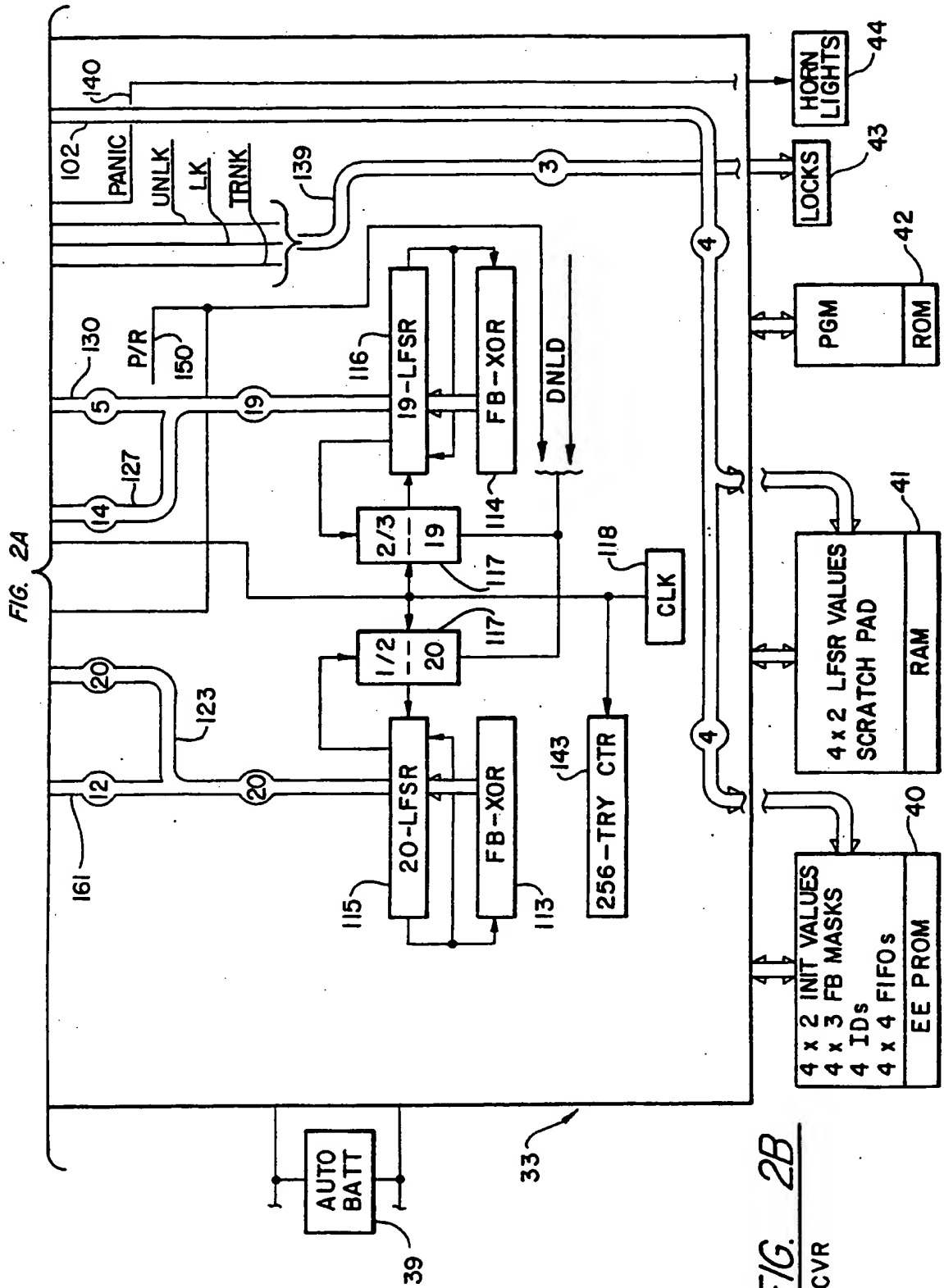


FIG. 2A
140
102
PANIC
UNLK
LK
TRNK
139
130
P/R
150
116
19-LFSR
FB-XOR
DNLD
127
2/3
19
114
117
118
CLK
1/2
20
123
161
20-LFSR
FB-XOR
115
113
143
256-TRY CTR
HORN LIGHTS
44
LOCKS
43
PGM
ROM
42
4 x 2 LFSR VALUES
SCRATCH PAD
RAM
41
4 x 2 INIT VALUES
4 x 3 FB MASKS
4 IDs
4 x 4 FIFOs
EE PROM
40
AUTO BATT
39
33

FIG. 2B
RCVR

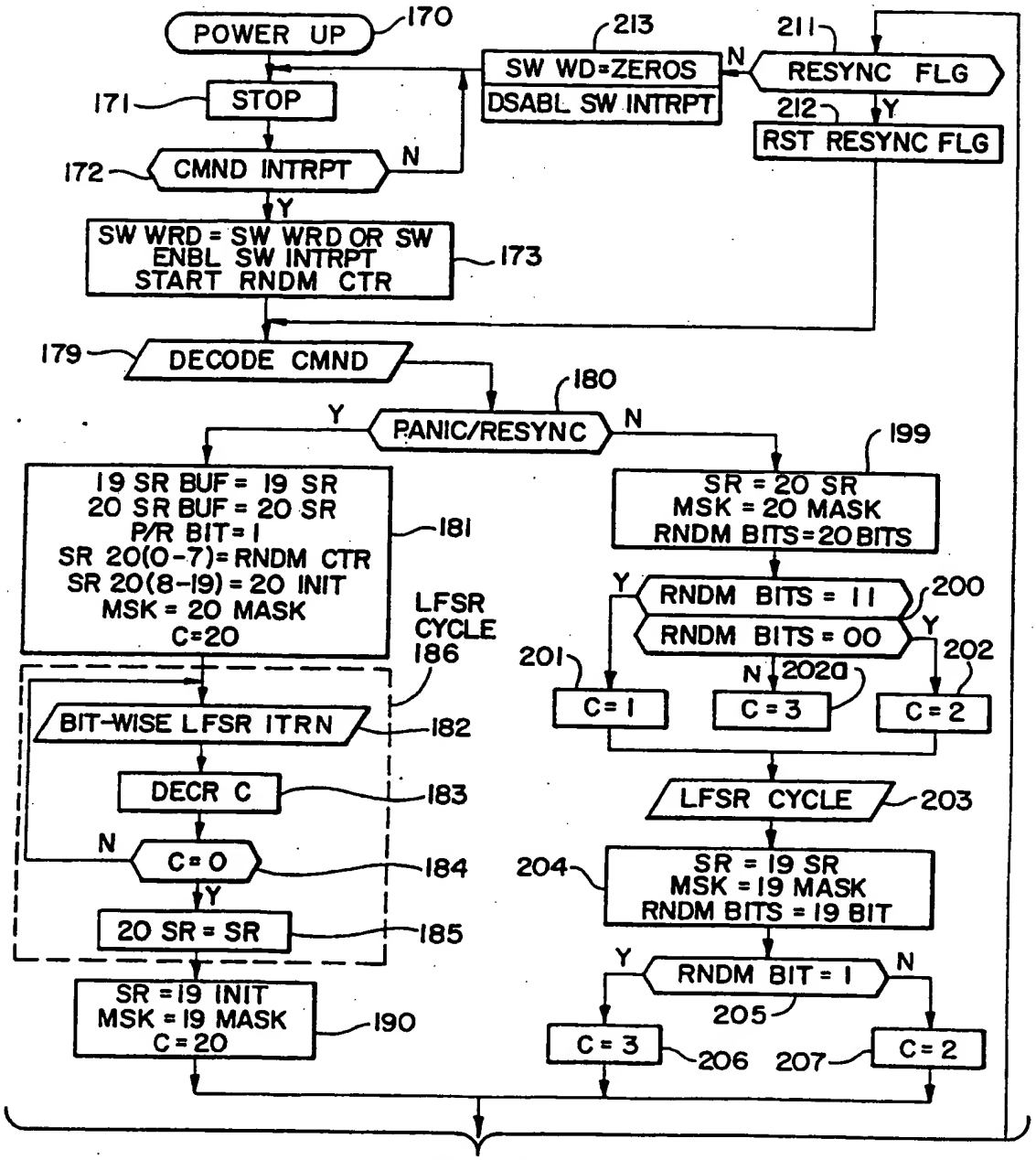


FIG. 3B

FIG. 3A

FOB - ENCRYP

FIG. 3A

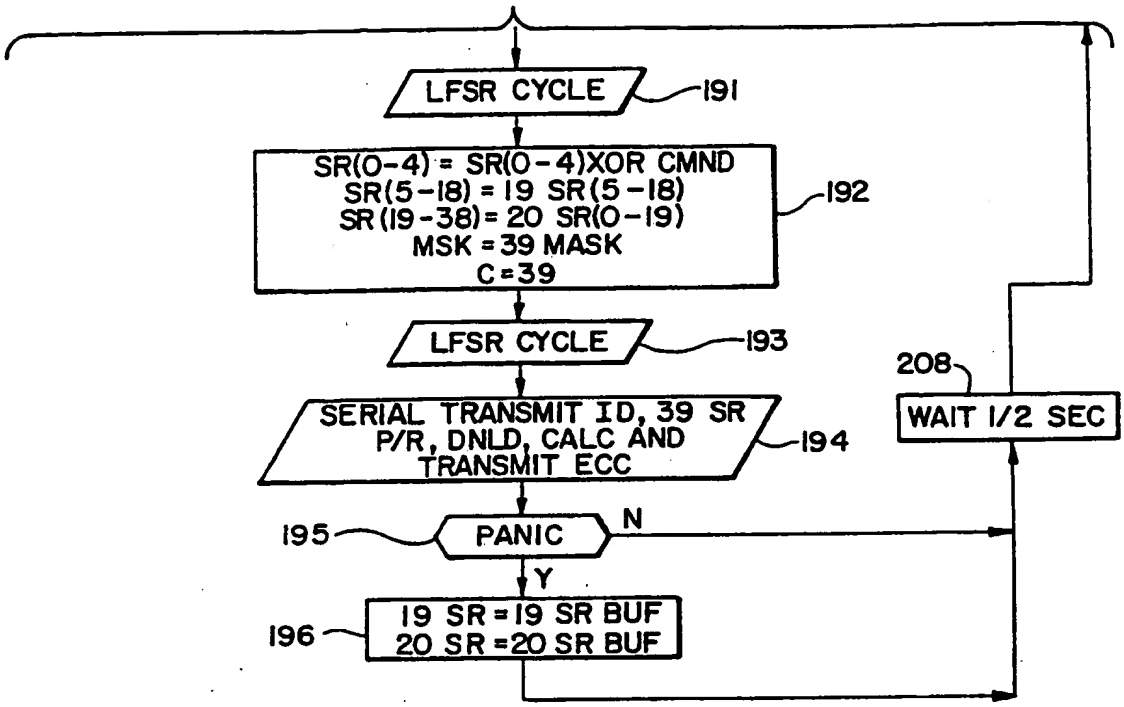


FIG. 3B

FOB - ENCRYP

FIG. 4

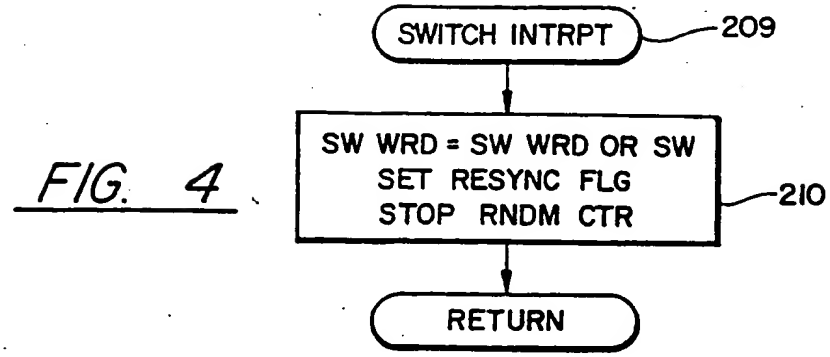


FIG. 5

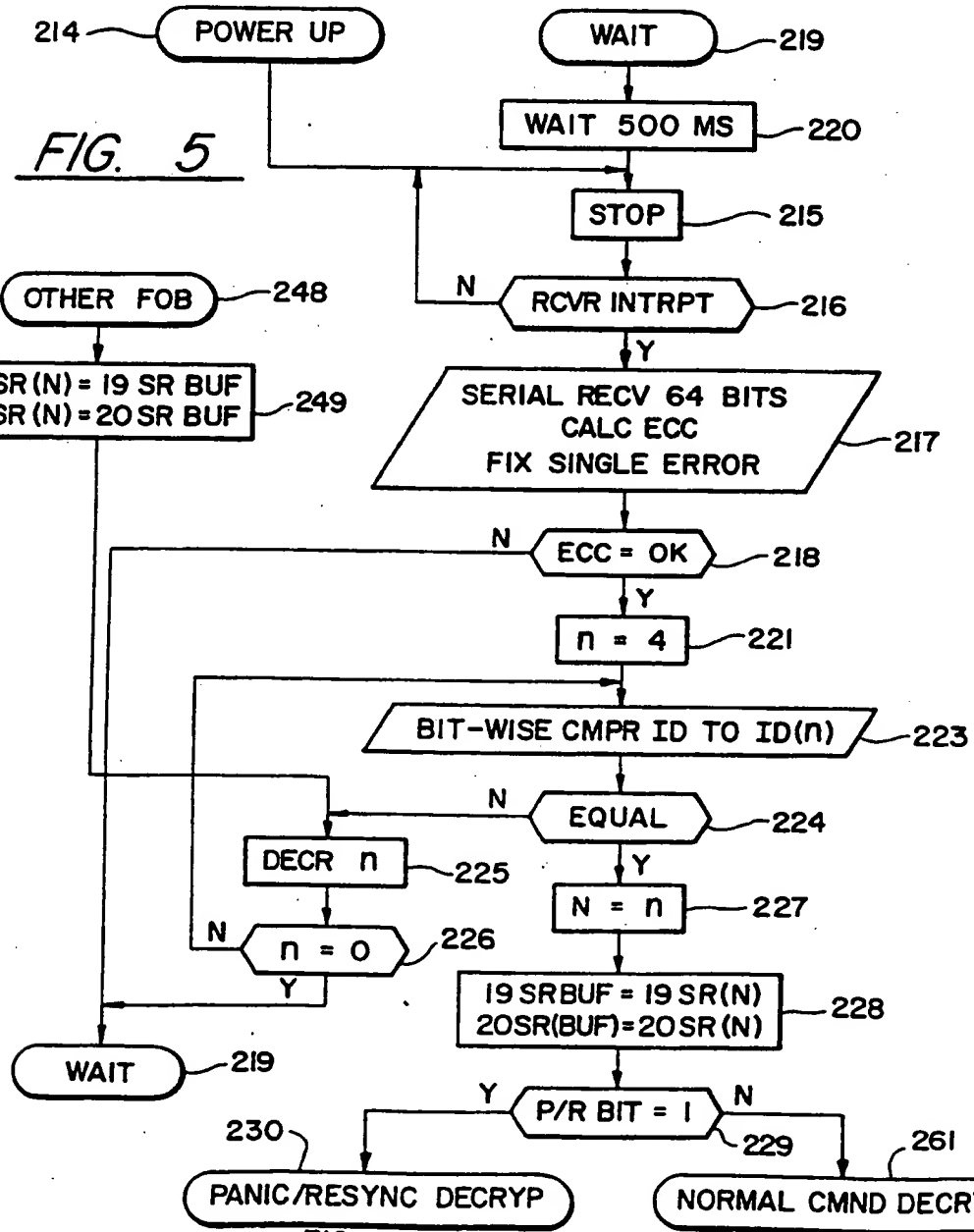


FIG. 6

FIG. 7

FIG. 6

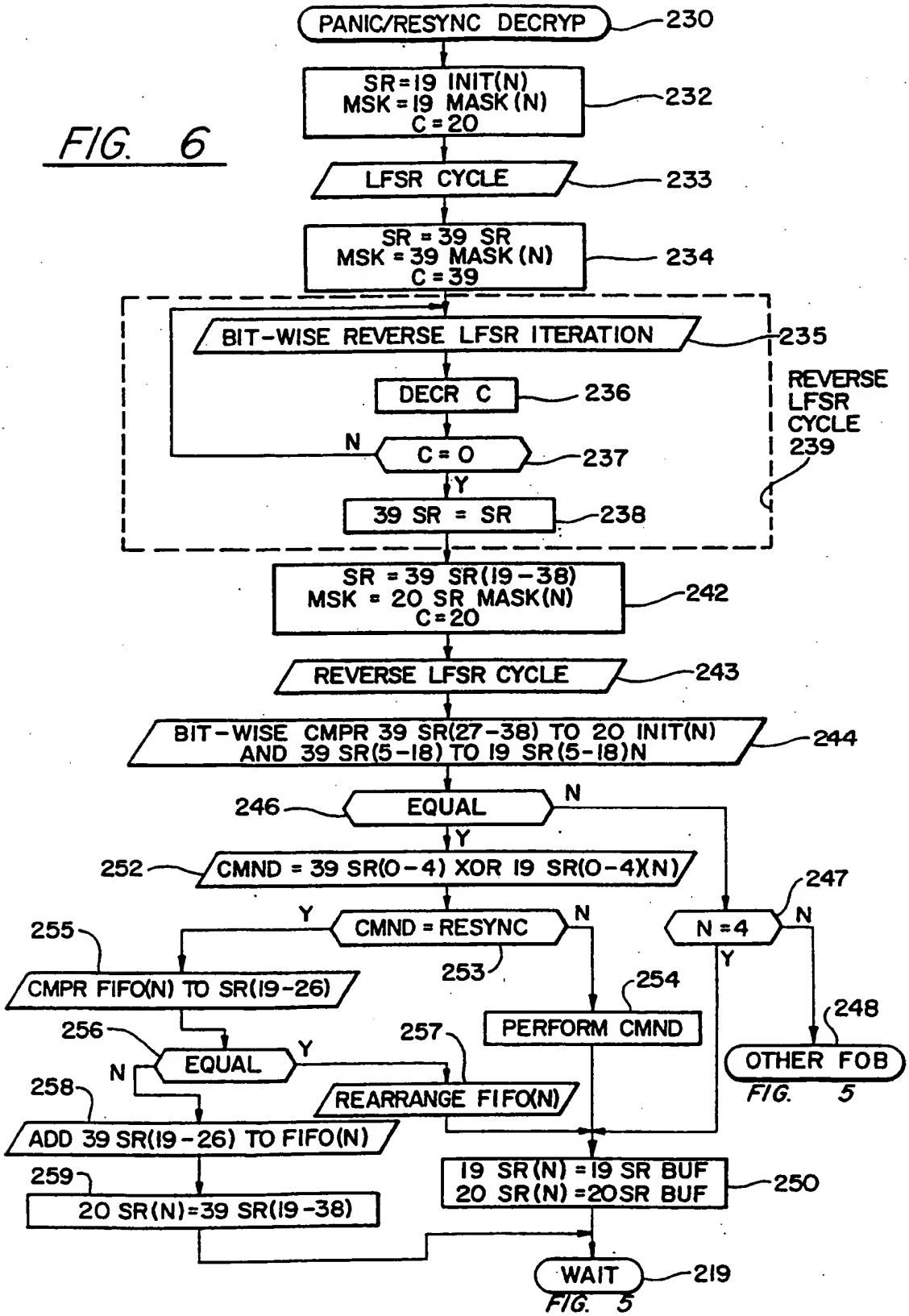
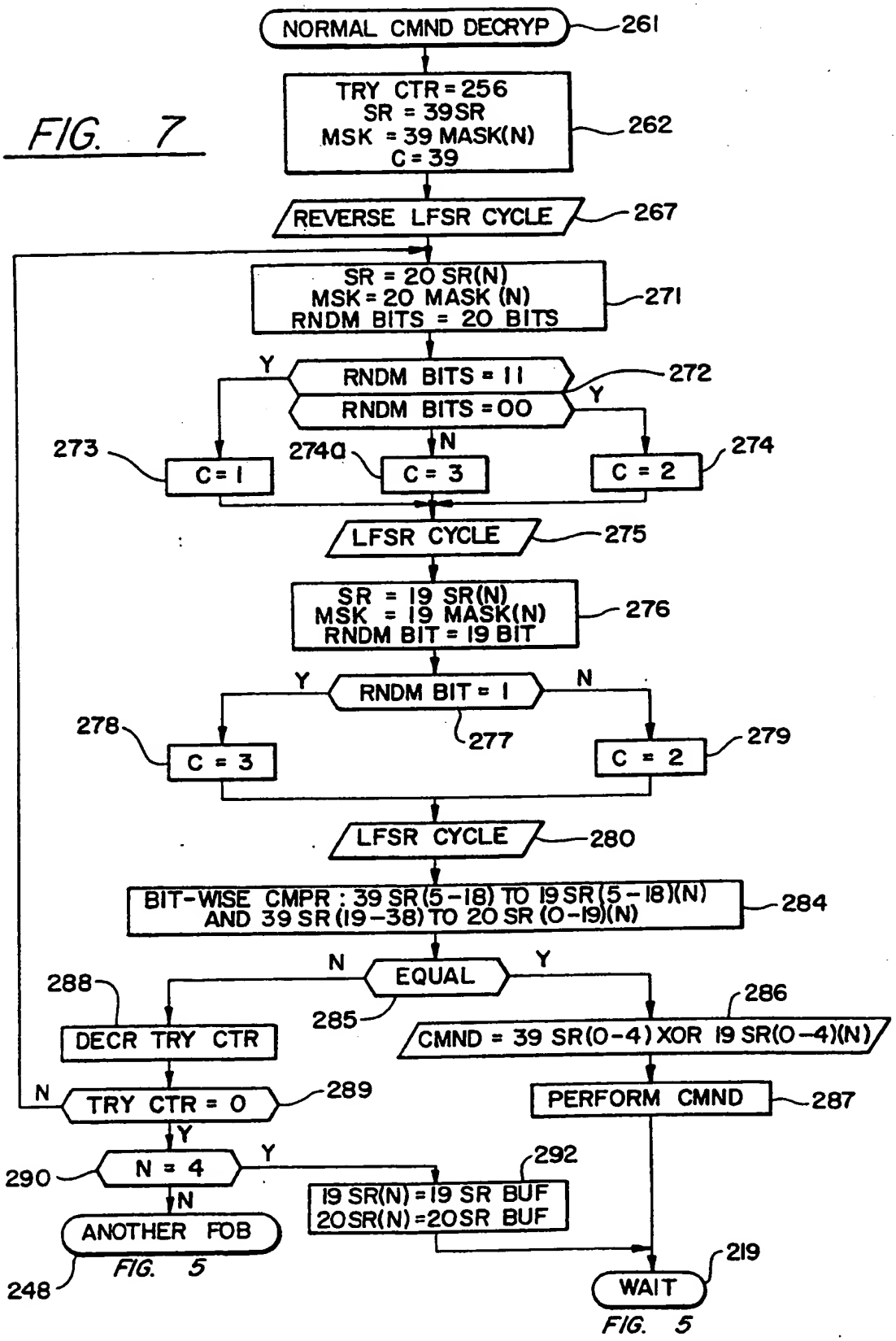

PANIC/RESYNC DECRYP
230
SR=19 INIT(N)
MSK=19 MASK(N)
C=20
232
LFSR CYCLE
233
SR=39 SR
MSK=39 MASK(N)
C=39
234
BIT-WISE REVERSE LFSR ITERATION
235
DECR C
236
N
C=0
237
Y
39 SR = SR
238
REVERSE LFSR CYCLE 239
SR=39 SR(19-38)
MSK=20 SR MASK(N)
C=20
242
REVERSE LFSR CYCLE
243
BIT-WISE CMPR 39 SR(27-38) TO 20 INIT(N) AND 39 SR(5-18) TO 19 SR(5-18)N
244
246
EQUAL
N
Y
252
CMND=39 SR(0-4) XOR 19 SR(0-4)(N)
247
Y
CMND=RESYNC
N
N=4
N
255
Y
253
CMPR FIFO(N) TO SR(19-26)
254
PERFORM CMND
248
OTHER FOB
FIG. 5
256
257
EQUAL
Y
N
REARRANGE FIFO(N)
258
ADD 39 SR(19-26) TO FIFO(N)
259
20 SR(N)=39 SR(19-38)
19 SR(N)=19 SR BUF
20 SR(N)=20SR BUF
250
WAIT
219
FIG. 5

FIG. 7


NORMAL CMND DECRYP
261
TRY CTR = 256
SR = 39 SR
MSK = 39 MASK(N)
C = 39
262
REVERSE LFSR CYCLE
267
SR = 20 SR(N)
MSK = 20 MASK (N)
RNDM BITS = 20 BITS
271
Y
RNDM BITS = 11
272
Y
RNDM BITS = 00
N
273
C = 1
2740
C = 3
C = 2
274
LFSR CYCLE
275
SR = 19 SR(N)
MSK = 19 MASK(N)
RNDM BIT = 19 BIT
276
Y
RNDM BIT = 1
N
277
278
C = 3
C = 2
279
LFSR CYCLE
280
BIT-WISE CMPR : 39 SR(5-18) TO 19 SR(5-18)(N)
AND 39 SR(19-38) TO 20 SR (0-19)(N)
284
N
EQUAL
Y
288
285
286
DECR TRY CTR
CMND = 39 SR(0-4) XOR 19 SR(0-4)(N)
N
TRY CTR = 0
289
PERFORM CMND
287
Y
290
N = 4
Y
292
N
19 SR(N) = 19 SR BUF
20 SR(N) = 20 SR BUF
ANOTHER FOB
FIG. 5
248
219
WAIT
FIG. 5

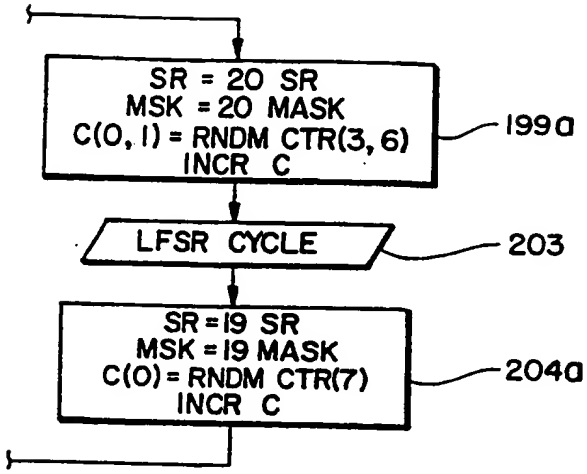

SR = 20 SR
MSK = 20 MASK
C(0, 1) = RNDM CTR(3, 6)
INCR C
199a
LFSR CYCLE
203
SR = 19 SR
MSK = 19 MASK
C(0) = RNDM CTR(7)
INCR C
204a

FIG. 8

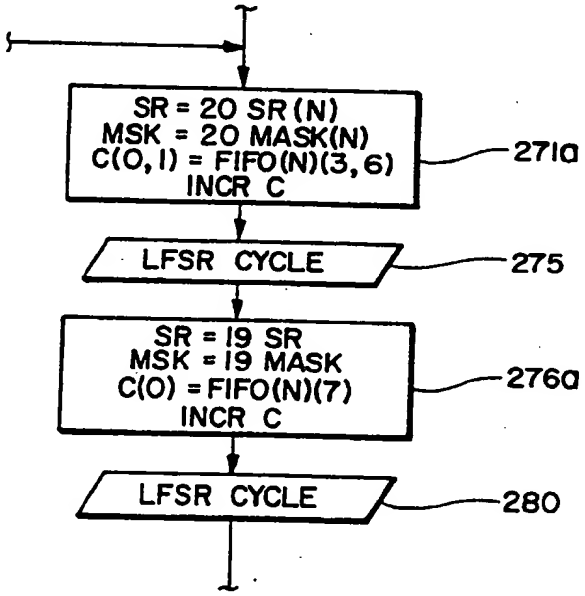

SR = 20 SR(N)
MSK = 20 MASK(N)
C(0, 1) = FIFO(N)(3, 6)
INCR C
271a
LFSR CYCLE
275
SR = 19 SR
MSK = 19 MASK
C(0) = FIFO(N)(7)
INCR C
276a
LFSR CYCLE
280

FIG. 9